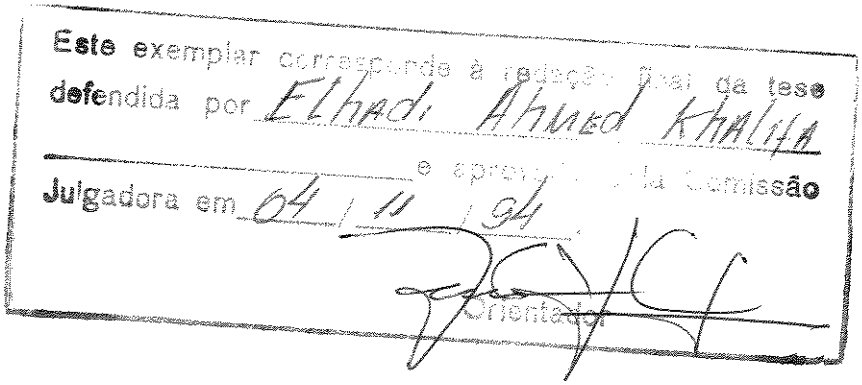
UNIVESIDADE ESTADUAL DE CAMPINAS
FACULDADE DE ENGENHARIA ELÉTRICA
DEPARTAMENTO DE COMUNICAÇÕES



CODIFICAÇÃO E DECODIFICAÇÃO DE ARQUIVOS DE FACSIMILE - G3/CCITT

Autor : ELHADI AHMED KHALIFA
Orientador: PROF. DR. JOÃO YABU-UTI

Tese apresentada à Faculdade de Engenharia Elétrica da Universidade Estadual de Campinas - UNICAMP - como parte dos requisitos exigidos para obtenção do titulo de MESTRE EM ENGENHARIA ELÉTRICA

OUTUBRO 1994

O PROFETA MUHAMMAD,(paz esteja com ele), Disse:

" Se alguem trilha um caminho em busca de SABEDORIA ,ALLAH tornar suave seu caminho para PARAISO." (MUSLIM).

E ainda disse:" A procura de SABEDORIA é obrigação de tod muçulmano." (IBN MAJAH)

Ã minha familia,

Meu pai, Ahmed Ali Khalifa

Minha mãe, Fayza Mohamed Khalifa

Minha querida esposa Dirlene e meus queridos filhos Youssef e Fayza pelo apoio em todos os sentidos e pela compreenção.

AGREDECIMENTO

Meus agredecimentos a todo poderoso ALLAH que me deu apoio, saude, orientação e força para finalizar esse trabalho.

A Prof. Dr. João Yabu-uti pelo dedicação , orientação e auxilio prestado.

A Prof.Afonso Alonso e Marcus Ponce pelo apoio e a valiosa colaboração para finalizar esse trabalho, os meus sinceros reconhecimento e gratidão para eternidade.

A Prof. Dr. Yuzo Iano pelo apoio moral em todos os sentidos.

A Sr. Suliman Al-Rajehi, e Dr. Ahmed Totwanji (JASIP) pelo apoio e ajuda para finalizar esse trabalho.

A Abdalla Ibrahim Elusta pelo ajuda e concelho de amigo prestado.

A todos os amigos que acompanharam-me nessa trajetória a minh eterna lembrança em particular :

a) Americo e Casmine
b) Isamara Carvalho
c) Marcelo Moreira
d) Edgard Luciano Oliveira da Silva
e) desenhistas Airton Ramos e Lucia
f) Marcia Maria Miranda

ÍNDICE

CAPÍTULO I

INTRODUÇÃO 1

CAPÍTULO II

HISTÓRICO DE DESENVOLVIMENTO DO FACSÍMILE E REGRAS DA CCITT . . 6

CAPÍTULO III

DESCRIÇÃO DAS FERRAMENTAS DE TRABALHO 25

CAPÍTULO IV

CODIFICAÇÃO 43

CAPÍTULO V

DECODIFICAÇÃO 62

CAPÍTULO VI

COMENTÁRIOS, RESULTADOS FINAIS E CONCLUSÕES FINAIS 81

BIBLIOGRAFIA 88

APÊNDICE.....

A LISTAGEM COMPLETA DO PROGRAMA DE CODIFICAÇÃO.....

B LISTAGEM COMPLETA DO PROGRAMA DE DECODIFICAÇÃO.....

C LISTAGEM COMPLETA DO PROGRAMA BASIEX.....

CAPíTULO I - INTRODUÇÃO
Introdução....1
I.1 Motivos para o uso de Fac-Símile....2
I.2 Descrição Geral do Projeto de Tese....3
CAPíTULO II -HISTóRICO DE DESENVOLVIMENTO DO FAC-SIMILE E REGRAS DA CCITT
II.1 Aparelho do Fac-símile....6
II.2 Regras do CCITT....11
II.3 Resumo Coparativo Entre G1,G2,G3 e G4....20
II.4 Perspecctivas Técnicas dos Aparelhos do G3 no Futuro....23
II.4.1 Fac-símile de Cor Completo....24
II.4.2 Fac-símile de Multicores....24
CAPíTULO III - DESCRIÇÃO DAS FERRAMENTAS DE TRABALHO
III.1 Linguagem de Máquina....25
III.2 Linguagem Assembly....27
III.3 Linguagem de Alto Nível....29
III.4 Programa de Auxílio usando BASIEX....30
III.5 Microcomputador....32
III.6 Impressora....34
III.7 Programa de depuração (DEBUG)....35
III.8 Hardware do PC-XT....39
III.9 Estrutura Interna do Microcomputador....40
III.10 Definições....40
CAPíTULO IV - CODIFICAÇÃO
IV.1 Introdução....43
IV.2 Descrição Geral do Processo de Codificação....45
IV.3 Rotina de Inicialização....48
IV.4 Rotina T....50
IV.5 Rotina CCD....51
IV.6 Rotina Mais....52
IV.7 Rotina Preto e Branco....53
IV.8 Rotina Trans....54
IV.9 Rotina R1....55
IV.10 Rotina Offset....55
IV.11 Rotina FDL....56
IV.12 Rotina A....56
IV.13 Rotina Resolve....57
IV.14 Rotina Bufcod....57
IV.15 Rotina Emenda....58

IV.16 Rotina Fcod 59
IV.17 Rotina PM 59
IV.18 Rotina Compos 60
CAPíTULO V - DECODIFICAÇÃO
V.1 Descrição Geral das Rotinas de Decodificação 62
V.2 Rotina de Inicialização 65
V.3 Rotina Consu 68
V.4 Rotina Term 69
V.5 Rotina Começo 69
V.6 Rotina Mudar 70
V.7 Rotina Zerab 71
V.8 Rotina Zero/FDL 71
V.9 Rotina Pegmb 72
V.10 Rotina EXFDL 73
V.11 Rotina ERRO 75
V.12 Rotina MONBUF 77
V.13 Rotina PRNT 78
CAPíTULO V - RESULTADOS E CONCLUSÕES
VI.1 Análise dos Códigos de Huffman Modificado 81
VI.2 Análise dos Resultados da Impressão 83
VI.3 Conclusão 86
VI.4 Propostas para Trabalhos Futuros 87
BIBLIOGRAFIA 88
APÊNDICE 89
A Listagem Completa do Programa de Codificação 90
B Listagem Completa do Programa de Decodificação 91
C Listagem Completa do Programa HUFCOD 92

CAPÍTULO I - INTRODUÇÃO

Neste trabalho , são abordados os tópicos relacionados com o equipamento de fac-símile de documentos que pode ser usado na rede telefônica nacional ou internacional, dando ênfase ao "software" do proceso de codificação e de decodificação do sistema de fac-símile digital do Grupo 3 .

O presente trabalho consiste de seis capítulos,ou seja, apresenta-se no capítulo :

I -Uma breve descrição do conteúdo deste trabalho (projeto).

II -Um histórico do equipamento, além de alguns aspectos relativos às regras de padronização do fac-símile exigidas pela Comissão Mundial em Telefonia e Telegrafia (CCITT).

III-Os princípios gerais de funcionamento das ferramentas de trabalho utilizadas neste projeto, como por exemplo, o microcomputador, hardware e software, programas, impressoras, etc.

IV -As análises e descrições de várias rotinas de assembler ligadas ao processo de codificação de uma imagem de fac-símile original.

V -As análises e descrições de várias rotinas de assembler ligadas ao processo de decodificação de uma imagem de fac-símile codificada.

VI -Os resultados obtidos deste trabalho e as conclusões finais sobre os processos de codificação e decodificação de imagens de fac-símile por microcomputador. Além disso, apresenta-se algumas perspectivas técnicas e sugestões para a continuidade deste trabalho no futuro.

I.1 - Motivos para o uso de Fac-Símile

Porque o uso de fac-símile ? Há várias razões de uso do fac-símile no Brasil e no mundo, das quais algumas são citadas a seguir.

1) Utilização da máquina de FAX, basicamente para fazer com que documentos utilizados nas áreas técnicas, comerciais, particulares, etc cheguem de forma rápida aos destinatários, evitando o uso de meios convencionais, normalmente caros e lentos (malote, correio , etc).

2) Uso de FAX resulta numa sensível melhoria no fluxo de informações entre áreas críticas para o funcionamento geral de qualquer empresa pública ou privada, aumentando a eficiência de comunicação de qualquer estabelecimento público ou privado.

3) Uso de fac-símile também evita erros de interpretação de mensagens, muitas vezes frutos de conversas telefônicas informais. Palavras podem ser esquecidas quando ditas durante uma conversa, mas nunca se estiverem documentadas. Uma tarefa é melhor entendida se for visualizada e descrita por palavras, ao passo que, por telefone, ela pode resultar numa demora desnecessária. Assim, o uso do Fax implica em economia de tempo e dinheiro.

4) Também auxilia nas comunicações em outras línguas estrangeiras visto que um documento escrito é bem mais fácil de ser entendido (com ajuda de um bom dicionário da própria língua) e não se gastaria muito tempo durante a comunicação.

5) O tempo necessário para ler uma página ao telefone é mais lento do que mandar a mesma página pelo Fax.

1.2 - Descrição Geral do Projeto de Tese

A idéia básica do projeto é utilizar um microcomputador como um aparelho de fac-símile, atribuíndo-lhe mais um serviço.

Uma página de tamanho A4 (210mm x 297mm) pode conter informações escrita, manuscrita, figuras, diagramas, mapas ou qualquer outra informação gráfica. Neste projeto esta página é dividida verticalmente em 1147 linhas de informações gráfica e são consideradas apenas dois tons, preto e branco, embora, seja possível usar vários tons de uma graduação de cinza, desde branco até preto. Para se ter uma destas variações, pode-se ter até 256 tons codificando os níveis de cinza com 8 bits.

Cada linha, no caso de dois tons, é formado por uma corrente de pontos pretos e/ou brancos e é varrida horizontalmente por um sensor óptico linear. Este sensor tem 2048 pontos sensíveis à luz. Assim, cada linha deinformação é constituída por 2048 pontos brancos e/ou pretos. Estes pontos são transferidos para a memória do microcomputador através de uma interface serial ou paralela para identificação e processamento de cada linha original de informação.

As correntes de pontos pretos e brancos são separadas e contadas no programa de codificação, e a cada uma delas é atribuída um código, de acordo com um processo de codificação chamado de "Código de Huffman Modificado" (CHM). Este código aproveita-se da estatística de ocorrência de cada corrente de pontos para diminuir a taxa de transmissão, atribuíndo códigos curtos às correntes com maior frequência de ocorrência e códigos copridos às correntes de pontos com menor frequência de ocorrência. Assim, em média, menos bits são atribuídos para cada corrente de pontos transmitido.

O programa de codificação usa um algorítmo do código CHM para codificar as informações provenientes do sensor óptico (CCD). Após a codificação das linhas originais de cada página, o programa agrupa e agrupa e organiza os códigos numa forma compactada e adequada para a transmissão.

No processo de decodificação dos códigos recebidos por um outro microcomputador, de início, os códigos vão para uma área de memória do microcomputador chamada de "buffer "de recepção. Em seguida, o programa de decodificação faz a leitura dos primeiros códigos, bit a bit. Após formar os primeiros códigos, o programa consulta duas tabelas de códigos de pontos pretos e brancos. Depois de achada a corrente de pontos correspondente ao código, o programa monta um buffer de impressão onde as correntes de pontos são organizadas de forma compacta e adequada para a impressão da imagem recebida. O buffer de impressão é organizado para conter 8 linhas de 2048 pontos, e assim, a cabeça de impressão pode imprimir 8 linhas simultaneamente acionando as 8 agulhas numa única varredura. Após a organização do buffer de impressão, começa o processo de impressão de imagem recebida onde é usada uma impressora para reproduzir a imagem original. A impressora é configurada para funcionar no modo gráfico a fim de conseguir a resolução de impressão desejada.

Neste trabalho, a idéia é usar a própria impressora para varrer o documento original, fixando a sensor óptico em cima do cilindro de impressão e empregando as instruções de avanço de micro-espaçamento a fim de varrer, linha por linha, o documento original.

No início deste projeto, esperou-se obter uma imagem de reprodução de alta resolução (2048 pontos por linha horizontal e 1147 linhas na vertical). Embora seja possível captar uma imagem com esta alta resolução usando um sensor óptico de 2048 elementos de imagem, no decorrer desta tese, foram detectados vários problemas para alcançar o objetivo em função dos equipamentos de trabalho disponíveis.

Por exemplo, a impressora RIMA disponível apresenta o cursor da cabeça de impressão pequeno para os 2048 pontos. Consequentemente, a cabeça de impressão preenche uma linha inteira em uma folha grande e retorna para o início da mesma linha para imprimir o restante dos pontos. Outro problema com a impressora disponível é o tamanho das agulhas de impressão que é excessivamente grande para impressão em alta resolução. Este aspecto faz com que os pontos de impressão fiquem muito próximos, a ponto de desaparecer um ponto branco existente entre

dois pontos pretos numa sequência. Assim, a definição da imagem impri-mida é seriamente prejudicada, tornando difícil o reconhecimento da imagem reproduzida.

A máxima resolução que a impressora RIMA permite para uma página de tamanho A_4 (297 mm x 210 mm) é de 504 pontos na direção horizontal e de 864 pontos na direção vertical. A distorção devida aos 360 pontos a mais na direção vertical pode ser resolvida selecionando uma adequada instrução de microespaçamento a ser usada para a impres-são. Desta forma, é possível obter as imagens de FAX aceitável com o esquema de codificação e decodificação proposto neste trabalho.

Capítulo II - Histórico de Desenvolvimento do Fac-Símile e Regras da CCITT

II.1 - Aparelho do Fac-símile

O aparelho de fac-símile moderno ou simplesmente chamado de "FAX" é um dispositivo que varre um documento e converte as informações gráficas em sinais elétricos. Estes, a seguir, são transformados em dados digitais para serem codificados e transmitidos para qualquer destino por redes nacionais ou internacionais de telefonia, via modem.

Um processamento inverso no aparelho receptor recupera as informações gráficas transmitidas. Uma vez que os sinais de fac-símile são codificados segundo uma padronização internacional (II.2), é possível enviá-los para qualquer parte do mundo, com exceção para os países que não permite a recepção de sinais não decodificados via rede telefonia, por motivos de segurança nacional (China,...).

O aparelho de fac-símile moderno é inteiramente digital e permite que um documento possa ser mandado em menos tempo do que ser descrito por uma conversação telefônica a um custo mínimo, mesmo para para chamadas urbanas e internacionais.

O aparelho de fac-símile foi e continua sendo uma ferramenta comum de produtividade para jornais, revistas, e escritórios de grande négocios. O aparelho era caro e difícil de mante-lo em funcionamento, e consequentemente só as empresas de comunicação mantiam um aparelho de fac-símile. Porém, graças ao avanço tecnológico da eletrônica digital, o FAX tornou-se um aparelho eletrodoméstico comum que permite uma boa reprodução do documento original, de modo simples, confiável e a preços baixos.

Para exemplificarmos a sua utilidade, vamos supor que você trabalha em sua casa e, certa vez, viajou para uma outra cidade a fim de concluir um negócio. Chegando lá, você descobre que precisa de alguns papéis que ficaram em casa e, neste caso basta pedir para alguém de sua casa para enviar os papéis via fac-símile. Em poucos

minutos e a custo bastante acessível, você terá em mãos as cópias do documentos necessários. Em uma outra situação, vamos supor que você está com problemas com a sua agência bancária, ou seja, ela o acusa de não ter pago uma conta. O problema pode ser facilmente resolvido em poucos minutos por um simples envio via FAX do documento que comprova o pagamento da conta à agência, ao invés de enviar uma cópia xerox do comprovante de pagamento via correio.

Nos próximos parágrafos, descreve-se uma versão antiga dos primeiros aparelhos de fac-símile no mundo. Na figura II.1, é mostrada uma representação gráfica de um aparelho de fac-símile antigo.

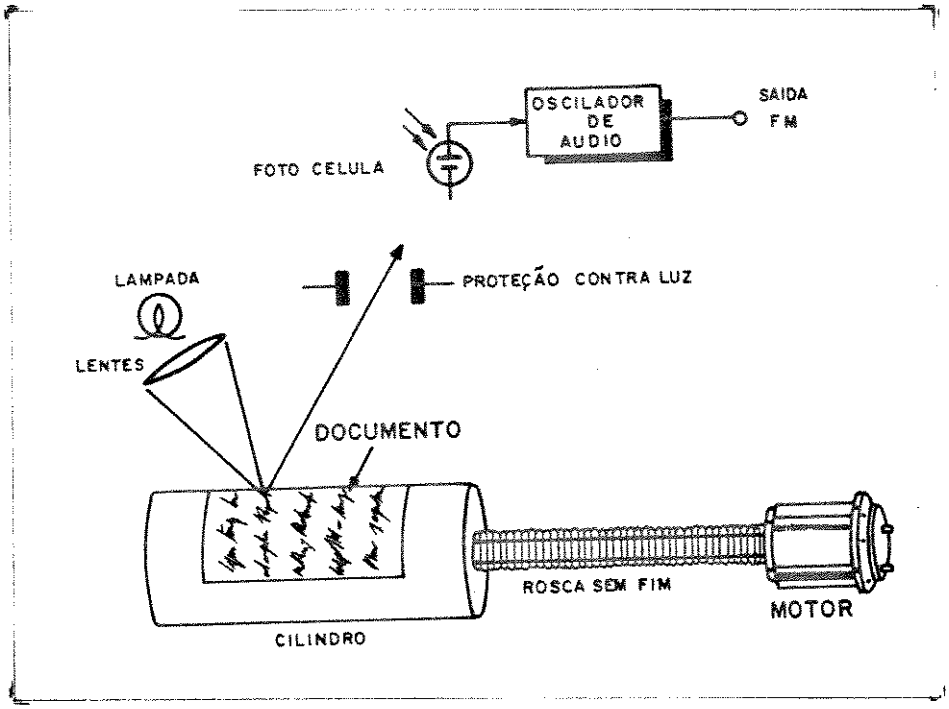


Fig.II.1 - Aparelho FAX de transmissão antigo

Neste aparelho, o documento a ser transmitido é colocado sobre um cilindro metálico que está conectado a um motor através de uma rosca sem fim. Quando o motor gira, o cilindro se desloca ao longo do percurso de rosca sem fim. Para transformar o documento em sinal elétrico, está posicionado acima do cilindro um sistema óptico constituído por uma lâmpada, uma lente para focalizar a fonte de luz e uma célula fotovoltaica conectada a um oscilador de áudio. Assim,enquanto o cilindro gira, o documento é varrido por um raio de luz e uma célula fotovoltaica capta a mudança da intensidade de luz causada pelos caracte-

res do documento. Esta variação luminosa produz uma variação na voltagem de corrente contínua correspondente que, por sua vez, aciona um oscilador de áudio controlado por voltagem. Como a frequência do oscilador é determinada pela voltagem gerada pela célula foto-voltaica, o sistema é chamado de modulação em frequência (FM).

O receptor é análogo ao sistema usado na transmissão, como mostrado na figura II.2. Neste sistema, a célula foto-voltaica e a fonte de luz são substituídas por uma agulha eletroestática. O demodulador de recepção converte os tons de áudio em sinais elétricos que acionam uma agulha eletroestática ou térmica para queimar um papel eletroestático ou térmico, reproduzindo assim a imagem do documento original.

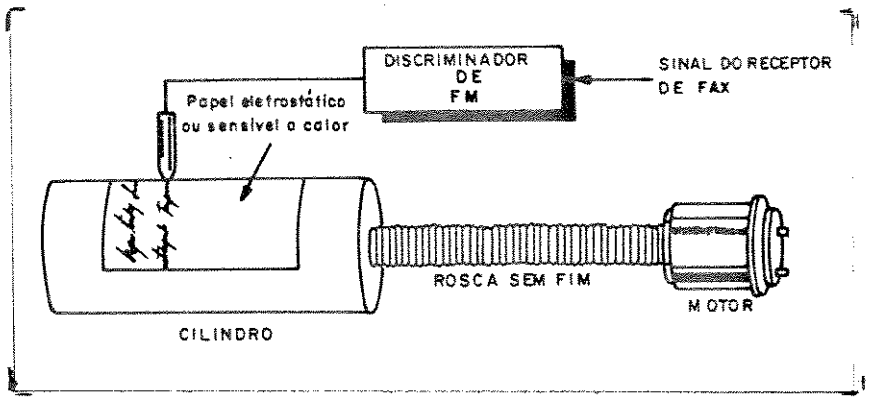


Fig.II.2 - Aparelho FAX de recepção antigo

A sincronização entre entre os cilindros de transmissão e de recepção tem sido obtida, em geral, usando motores de corrente alternada (AC) síncrona onde a velocidade dos motores é controlada pela frequência da rede elétrica. Quando se usa motores de corrente contínua (DC), muitos ajustes finos são necessários para a sincronização.

O sistema de cilindros foi usado por muitos anos, porém com o avanço da eletrônica digital, os cilindros mecânicos foram substituídos por varredura eletrônica. Assim, em aparelhos de fac-símile modernos, a sincronização é feita através de um sinal eletrônico,

reduzindo o tempo de transmissão, por página, de seis minutos para um minuto. Alguns aparelhos de FAX digitais chegam a transmitir uma página inteira em 15 a 18 segundos, além de permitir aos usuários outros tipos de serviços como armazenar e transmitir posteriormente as mensagens. A figura II.3 ilustra um diagrama de blocos de um aparelho de fac-símile de alto desempenho.

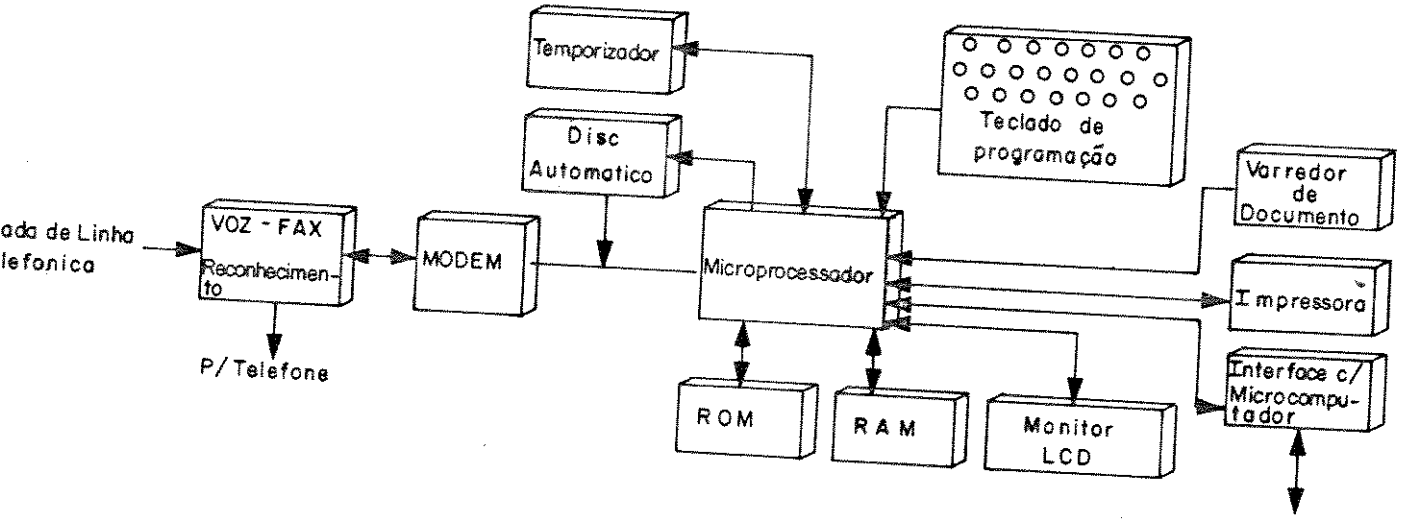


Fig.II.3 - Aparelho FAX moderno

Atualmente, os aparelhos de FAX estão mais compactos pela extensa utilização de circuitos integrados dedicados e pode ser utilizado como uma placa periférica de microcomputador PC-XT ou PC-AT, além de permitir uma ligação direta com linha telefônica através de um modem embutido na própria placa do FAX. A placa de fac-símile digital de alto desempenho normalmente contém programas que ficam na memória do microcomputador, tornando o aparelho mais ágil. Assim, a função do FAX é automática e disponível mesmo durante o tempo em que o microcomputador estiver sendo usado para outras tarefas. Esta placa oferece tudo que um aparelho de FAX convencional pode oferecer, além de poder armazenar textos gráficos e liberar dados contidos em um arquivo de disco. Em outras palavras, a placa de FAX pode transmitir dados de qualquer disco que está formatado como um sinal de fac-símile convencional e pode armazenar vários arquivos de FAX em um disco flexível ou rígido para futuras consultas ou impressões.

As opções de serviços de fac-símile são facilmente implementadas com programas, tais como armazenar páginas de multi-endereços, armazenar e enviar documento em um modo seletivo ou difusão e responder à aceitação e seleção do pedido. Essas opções são inúmeras e depende somente do grau de sofisticação dos programas implementados às custas de serviços mais caros.

II.2 - REGRAS DO CCITT

CARACTERÍSTICAS DAS MÁQUINAS DE FAC-SÍMILE

As máquinas de fac-símile classificam-se em quatro categorias ou grupos que definem as características de cada tipo de máquina. A seguir, descreve-se as características de cada grupo.

1) GRUPO UM (1)

Os aparelhos deste grupo usam modulação AM tipo banda lateral dupla ("double sideband"), sem quaisquer medidas especiais de compressão de largura de banda do sinal transmitido. Este aparelho é capaz de transmitir um documento de tamanho ISO A4, com velocidade nominal de 4 linhas/mm em quase seis (6) minutos, via linha telefônica. é possível operá-lo também com baixa resolução, adequada para transmissão de documento tamanho ISO A4 num intervalo de tempo entre 3 (três) e 6 (seis) minutos.

2) GRUPO DOIS (2)

Os aparelhos deste grupo utilizam técnicas de compressão de largura de banda, para conseguir um tempo de transmissão de quase três minutos, para um documento de tamanho ISO A4 com resolução de 4 linhas/mm. A transmissão é feita pela linha telefônica. A compressão de largura de banda neste contexto inclui codificação e/ou modulação VSB (banda lateral vestigial), mas não inclui o processamento de sinal do documento para reduzir a redundância.

3) GRUPO TRÊS (3)

Os aparelhos deste grupo usam técnicas de compressão de largura de banda e redução de informação redundante antes de efetuar a modulação do sinal de documento. Estes aparelhos podem transmitir um documento de tamanho ISO A4 em 1 minuto. A transmissão do documento é feita através da linha telefônica.

4) GRUPO QUATRO (4)

Os aparelhos deste grupo usam técnicas para reduzir a redundância da informação antes de transmitir os sinais do documento. A transmissão é feita através de Redes Públicas de Dados (PDN). Este aparelho, além de utilizar procedimentos aplicáveis a PDN também garante a recepção de documentos sem erros e pode ser usado nas redes públicas de telefonia, utilizando um processo adequado de modulação.

ALGUMAS DEFINIÇÕES DE TERMOS IMPORTANTES USADOS NAS RECOMENDAÇÕES DE APARELHOS DE FAC-SÍMILE

1) PEL

É a abreviatura de "picture element" ou elemento de imagem.

2) ELEMENTO DE FOTO

Na transmissão,ele é a parte da área do documento original em que concide com o ponto de varredura em um dado instante com a mesma intensidade, sem distinção de detalhes que pode ser incluída. Na recepção, ele é a área de detalhes mais finos que pode ser reproduzida durante a impressão.

3) COLOCAÇÃO EM FASE

O receptor deve assegurar a coincidência do ponto médio do seu campo de varredura com o correspondente ponto do transmissor, para garantir o posicionamento correto da imagem durante a impressão. A esta coincidência entre os pontos médios denomina-se PHASING.

4) TAXA DE REPRODUÇÃO

é a taxa das dimensões lineares do documento reproduzido em relação às dimensões do documento original correspondente.

5) LINHA DE VARREDURA

É a área explorada pelo ponto de varredura durante uma varredura de um quadro, de um lado para o outro.

6) DECLIVE DE VARREDURA

É a distância entre as bordas ou cantos de duas linhas de varredura consecutivas.

7) DENSIDADE DE VARREDURA

É o número de PEL's por unidade de comprimento.

8) RESOLUÇÃO

É a medida da capacidade do aparelho definir detalhes da imagem. Nos aparelhos dos Grupos 3 e 4, a resolução horizontal é expressa em números de elementos de imagem por milímetro (pels/mm) e a resolução vertical é dada pelo número de linhas de varredura por milímetro.

9) INCLINAÇÃO

É um defeito de reprodução que ocorre quando a linha se desvia segundo um ângulo de 90 graus com respeito à direção de varredura. Em consequência, as linhas ficam inclinadas em relação à direção varredura por causa das diferenças entre as velocidades de varredura na transmissão e na recepção.

10) SINCRONIZAÇÃO

É o estabelecimento de frequências iguais de varredura no transmissor e no receptor.

PADRONIZAÇÃO DE APARELHOS DE FAC-SÍMILE

Os padrões de construção dos aparelhos de FAX são apresentados a seguir.

1) TRILHA DE VARREDORA

É o sentido de varredura do documento. No transmissor, a área da mensagem deverá ser varrida no sentido negativo. A orientação do documento em relação ao plano de varredura vai depender de sua dimensão e não tem consequências. Deve-se notar que, no aparelho de recepção, a varredura é feita no sentido negativo para obter uma

recepção positiva, e no sentido positivo para obter uma recepção negativa.

2) ÍNDICE DE COOPERAÇÃO

O índice normal é 352 que corresponde ao fator de cooperação de 1105. Um índice alternativo é 264 , o qual é preferido quando uma varredura menos densa for requerida ou quando as características dos circuitos exigem particularmente a combinação de rádio e circuitos metálicos, e que corresponde ao fator de cooperação de 829. As tolerâncias permitidas dos valores mencionados acima são de ± 1%

3) DIMENSÕES DOS APARELHOS

a) Em aparelhos de FAX com cilindro de varredura, os cilindros mais usados são de diâmetros de 66 mm, 70 mm e 80 mm. O fator de cilindro do aparelho transmissor não deve ser maior do que 2,4 ,enquanto que, do FAX receptor, não deve ser menor do que 2,4 . A largura [despositiva que "contir" informação de imagem ou que pode ser chamado "Setor morto"] não deve ultrapassar 15 mm. Uma tolerância de quase 3 % do comprimento total de uma linha de varredura é usada para "amarrar" a fase.

b) Em aparelhos com varredura do tipo "flat-bed", os comprimentos mais usados de linha de varredura são de 207mm, 220 mm e 276 mm, dos quais 15 mm não são usados na transmissão por causa da possibilidade do FAX receptor ser do tipo que usa cilindro. Assim, antes de transmitir o documento para o aparelho que usa um cilindro, é necessário garantir que o valor seguinte seja verdadeiro, ou seja, o comprimento dos documentos transmitidos é dado pelo comprimento total da linha de varredura cujo valor deve ser menor ou igual do que o fator de cilindro do aparelho receptor.

4) TAXA DE REPRODUÇÃO

Nos casos de aparelhos interconectados terem comprimentos de linha de varredura diferentes mas com mesmo índice de cooperação, a reprodução do documento original transmitido terá uma pequena diferença no tamanho devido à diferença de comprimento da linha de varredura entre os aparelhos na trasmissão e recepção.

5) VELOCIDADE DE ROTAÇÃO DO CILINDRO
FREQUENCIA DE LINHA DE VARREDORA

Em condições normais, as velocidades do cilindro com a mesma frequência da linha de varredura são de 60 e 90 rotações por minuto (rpm). A velocidade do transmissor deve ser mantida o mais próximo possível da velocidade nominal e, em qualquer caso, tem que estar na faixa de ± 10 partes por milhão (ppm) da velocidade nominal. A velocidade no receptor deve ser ajustável e a faixa de ajuste deve ser, no mínimo, de ± 30 ppm da velocidade nominal. Após a regulação das velocidades no transmissor e no receptor, a diferença absoluta da velocidade entre os dois não deve maior do que 10 ppm.

6) JUDDER

A estabilidade da velocidade durante uma rotação deve ser controlada de tal maneira que o máximo deslocamento da superfície do cilindro de uma posição média não ultrapasse por um quarto de "SCANNING PITCH" no índice normal de 352. Esse controle pode também ser mantido desde que o máximo ângulo de oscilação não ultrapasse 0,80 graus, medidos de uma posição média.

PADRONIZAÇÃO DAS MÁQUINAS DE FAC-SÍMILE/GRUPO 3 PARA TRANSMISSÃO DE DOCUMENTO

Nos aparelhos de fac-símile do Grupo 3 usados nas redes de linha telefônica, os circuitos internacionais devem ser projetados segundo os padrões listados a seguir.

1 - TRILHA DE VARREDURA

A área do documento que contém informações deve ser varrida na mesma direção no transmissor e no receptor, em relação ao plano vertical do documento. Os elementos de informação devem ser varridos na mesma direção de varredura, da esquerda para a direita, e o processo de varredura deve ser sempre para baixo em relação à linha anterior.

2 - DIMENSÕES DE APARELHOS:

As seguintes dimensões devem ser usadas:

a) Resolução padrão de (3,85 ± 1%) linhas/mm e resolução alta de de (7,7 ± 1%) linhas/mm (opcional), na direção vertical.

b) Uma linha deve conter 1.728 elementos pretos e brancos numa linha de varredura com um comprimento de (215 ± 1%) mm .

c) O recurso opcional estabelece 2.048 elementos pretos e brancos numa linha de (255 ± 1%) mm e 2.432 elementos pretos e brancos numa linha de (303 ± 1%) mm .

3 - ESQUEMA DE CODIFICAÇÃO

A linha de dados é composta por séries de palavras-códigos de comprimento variável onde cada palavra código representa uma corrente de elementos brancos ou de elementos pretos. As correntes de elementos pretos são alternadas. Cada linha de varredura tem 1.728 elementos de imagem que representam uma linha de varredura horizontal de 215 mm de comprimento.

Para garantir a sincronização das cores na máquina receptora, todas as linhas de dados devem começar com a palavra-código de corrente branca. Se o primeiro ponto da linha atual inicial for preto, então uma palavra-código de corrente branca de zeros vai ser enviada. As linhas de pontos devem ter, no máximo, 1728 elementos de imagem, cujos os códigos estão definidos na figura II.4.

As palavras-códigos podem ser de terminação ou de composição. Cada corrente é representada por uma palavra-código de terminação ou por uma palavra-código de composição seguida por uma palavra-código de terminação. As correntes com comprimentos que variam de 0 até 63 PEL's são codificadas com a própria palavra-código de terminação. As correntes com comprimentos que variam de 64 até 1.728 PEL's são codificadas inicialmente por uma palavra-código de composição que representa um comprimento de corrente menor ou igual do que o requerido. Depois esta palavra-código é seguida por uma palavra-código de terminação, que representa a diferença entre o comprimento da corrente requerido e comprimento da corrente de palavra código de composição. Note-se que há duas listas de palavras-código, uma para as correntes brancas e uma outra para correntes pretas.

Fig.II.4 - Tabela de Código de Huffman Modificado

CORRENTE BRANCA	PALAVRA CÓDIGO	CORRENTE PRETA	PALAVRA CÓDIGO
0	00110101	0	0000110111
1	000111	1	010
2	0111	2	11
3	1000	3	10
4	1011	4	011
5	1100	5	0011
6	1110	6	0010
7	1111	7	00011
8	10011	8	000101
9	10100	9	000100
10	00111	10	0000100
11	01000	11	0000101
12	001000	12	0000111
13	000011	13	00000100
14	110100	14	00000111
15	110101	15	0000011000
16	101010	16	0000010111
17	101011	17	0000011000
18	0100111	18	0000001000
19	0001100	19	0000011001111
20	0001000	20	0000011010000
21	0010111	21	0000011011100
22	0000011	22	0000011011111
23	0000100	23	0000010101000
24	0101000	24	0000000101111
25	0101011	25	0000000110000
26	0010011	26	0000011001010
27	0100100	27	0000011001011
28	0011000	28	0000011001100
29	00000010	29	0000011001101
30	00000011	30	0000011010000
31	00011010	31	0000011010001
32	00011011	32	0000011010010
33	00010010	33	0000011010011
34	00010011	34	0000011010010
35	00010100	35	0000011010011
36	00010101	36	0000011010100
37	00010110	37	0000011010101
38	00010111	38	0000011010110
39	00101000	39	0000011010111
40	00101001	40	0000011011100
41	00101010	41	0000011011101
42	00101011	42	0000011011010
43	00101100	43	0000011011011
44	00101101	44	0000010101000
45	00000100	45	0000010101001
46	00000101	46	0000010101010
47	00001010	47	0000010101011
48	00001011	48	0000011001000
49	01010010	49	0000011001001
50	01010011	50	0000010100010
51	01010100	51	0000010100011
52	01010101	52	0000000100100
53	00100100	53	0000000110111
54	00100101	54	0000000111000
55	01011000	55	0000000100111
56	01011001	56	0000000101000
57	01011010	57	0000001011000
58	01011011	58	0000001011001
59	01001010	59	0000000101011
60	01001011	60	0000000101100
61	00110010	61	0000001011010
62	00110011	62	0000001100110
63	00110100	63	0000001100111

4 - TEMPO DE TRANSMISSÃO POR LINHA CODIFICADA

Uma linha total codificada é composta pela soma de bits de dados, de "FILL" e de FDL. Para facilidade de manuseio dos métodos de impressão, vários tempos opcionais de transmissão da linha codificada são possíveis, além do tempo padrão de 20 ms por linha codificada para uma resolução padrão e resolução alta [alternativa um]. Na alternativa dois, o tempo mínimo de transmissão de uma linha codificada para uma resolução alta é a metade do tempo requerido para uma resolução padrão, ou seja, é de 10 ms em alta resolução e 20 ms em resolução padrão. Em qualquer caso, o tempo máximo de transmissão de uma linha codificada deve ser menor do que 5 minutos.

5 - MODULAÇÃO E DEMODULAÇÃO

As máquinas do Grupo 3 operando em redes telefônicas usam a modulação, o "scrambler" e os sinais de equalização e temporização definidos na recomendação V.27 da CCITT.

5A- O sinal de treinamento deve ser de seqüência comprimida com proteção contra o eco de quem fala.

5B- A taxa de sinalização de dados pode ser de 4.800 bits/s ou 2.400 bits/s.

6 - POTÊNCIA NA SAÍDA DO TRANSMISSOR

A potência média deve ser ajustável entre -15 dbm até 0 dBm, porém sem a possibilidade do operador ajustar a potência do equipamento (auto-ajustável).

7 - POTÊNCIA NA ENTRADA DO RECEPTOR

O aparelho receptor deve ter um controle de sensibilidade para que, quando o sinal recebido estiver na faixa de 0 dBm até -43 dBm, o operador possa ajustar a sensibilidade desejada.

8 - IMPLEMENTAÇÃO DE APARELHOS

A implementação destas máquinas não precisa sempre de um varredor físico de papel e/ou impressora. Se a imagem for gerada através de outros meios de comunicação, os sinais de entrada na rede telefônica devem ser idênticos aos outros sinais gerados pela entrada de papel.

II.3 - Resumo Comparativo Entre G_1, G_2, G_3 e G_4

Apresenta-se a seguir um resumo comparativo dos aparelhos de FAX dos grupos G_1, G_2, G_3 e G_4. Para todos os grupos são comparados os seguintes ítens :

a - Rede usada

b - Ligação em redes

c - Procedimento de controle

d - Natureza do sinal de fac-símile

e - Modulação

f - Resolução de impressão e varredura

g - Sincronização de linha

h - Tempo de transmissão por cada linha

Para os grupos G_3 e G_4 são comparados os seguintes ítens adicionais :

i - Taxa de transmissão digital

j - Tipo de codificação usada

k - Controle de erro

l - Estrutura de quadro

a - Rede Usada

Os grupos G_1 a G_3 usam a rede pública de telefonia e também faz ligações com outras redes via sistema de sinalização de rede pública de telefonia enquanto que o grupo G_4 usa principalmente a rede pública de dados (PDN).

b - Ligação em Redes

Os grupos G_1 a G_3 usam o sistema de sinalização da rede pública de telefonia (PSTN) enquanto que G_4 usa o procedimento de controle de PDN (X.21, X.25).

c - Procedimento de Controle

Os grupos G_1 e G_2 usam a sinalização tonal e o grupo G_3 usa a sinalização binária codificada, enquanto que G_4 usa o procedimento para teletexto e serviços de fac-símile de G_4.

d - Natureza do sinal de fac-símile

No G_1 e G_2, o sinal é puramente analógico enquanto que em G_3 e G_4, ele é digital.

e - Modulação

Tem-se as seguintes modulações para os grupos :

G1: modulação de amplitude (AM) ou modulação de frequência (FM).

G2: modulação AM, modulação de fase (PM) ou modulação de banda lateral vestigial (VSB).

G3: modulação digital diferencialmente codificados (8-PSK) para a taxa de 4800 bits/seg. ou modulação de quadratura de amplitude (QAM) para a taxa de transmissão de 9600 bits/seg.

G4: modulação especificada pela rede usada.

f - Resolução de Impressão e Varredura

No G_1 e G_2 usa-se uma resolução horizontal de 1728 pels por 215 mm e uma resolução vertical de 3,85 linhas por mm. No G_3 usa-se as mesmas resoluções vertical e horizontal de G_1 ou G_2, exceto em alta resolução que requer 7,7 linhas/mm na vertical. No grupo G_4 usa-se resoluções vertical e horizontal de 200, 300 ou 400 ppp.

g - Sincronização de Linha

Nos grupos G_1 e G_2, usa-se um sinal de portador de nível máximo enquanto que no grupo G_3 usa-se o sinal de fim de linha. No grupo G_4 não requer o sinal de sincronização pois, para tanto, usa-se o protocolo de comunicação de dados.

h - Tempo de Transmissão por Linha

No G_1 leva 1/180 minutos/linha e no G_2 leva 1/360 minutos/linha. No G_3, este tempo depende da taxa de compressão obtida, podendo atingir um tempo mínimo de 5 ms, 10 ms, 20 ms ou 40 ms. No G_4, esse tempo depende completamente da técnica de compressão que for utilizada.

i - Taxa de Transmissão Digital

No grupo G_3, para transmitir os dados de FAX mais o sinal de controle são necessários 2400, 4800, 7200, 9600 ou 48000 bits/seg., dos quais 300 bits/seg são usados para o controle. No grupo G_4, a taxa é de 64 Kbits/seg. para transmissão pela rede digital de serviços integrados (ISDN).

j - Tipo de Codificação Usada

No grupo G_3 usa-se o código de Huffman Modificado (CHM) para codificação unidimensional e código de leitura modificada (MR) com K = 2,4 para codificação bidimensional. No grupo G_4 usa-se a codificação bidimensional completa.

k - Controle de Erro

No G_3, usa-se a técnica de retransmissão de linha para controlar o erro enquanto que no G_4 usa-se a técnica HDLC para o controle de erro.

l - Estrutura de Quadro

No G_3, usa-se um quadro de HDLC apenas no sinal de controle (dados não estruturados) enquanto que no G_4 usa-se um quadro HDLC para sinais de dados e de controle.

Observando as diferenças entre os vários grupos de fac-símile, chega-se a uma conclusão de que, para os 4 grupos de FAX serem compatíveis, são necessárias algumas modificações. Por exemplo, o aparelho FAX de G_1 ou G_2 requer um conversor análogo-digital para compatibilizar com os dados do aparelho FAX de G_3 ou G_4. Existe uma relação enorme de modificações a serem efetuadas, caso queira que um documento transmitido por um aparelho FAX de um grupo seja recebido corretamente por um aparelho FAX do outro grupo.

II.4 - Perspectivas Técnicas dos Aparelhos do G_3 no Futuro

Existe uma grande variedade de modelos de aparelhos FAX do grupo G_3 , desde pequenos aparelhos de baixo custo com as funções básicas até aqueles de alto grau de sofisticação com diversas funções opcionais.

Do ponto de vista técnico da evolução de FAX do G_3, o projeto desses aparelhos está saíndo do cilindro mecânico de varredura para sistema eletrônico de varredura ("flat-bed") e os aparelhos estão se tornando menores em tamanho e mais rápido na transmissão.

Para conversão foto-elétrica, dentre os sensores foto-elétricos para transmissão, os sensores ópticos lineares (CCD) são os mais usados. Recentemente um tipo de sensor óptico de contacto físico com documento está começando a tomar espaço no mercado pois ele, entre outros aspectos,permite uma maior compactação do aparelho FAX.

No campo da reprodução de informação gráfica original no receptor, existem várias maneiras de reproduzi-la ou imprimi-la. Ela pode ser feita por impressão eletrostática, impressão térmica, jato de tinta spark (faísca), etc. Mas as impressões térmica e eletrostática são mais usadas [1] em fac-símile fotográfico. Estes tipos são principalmente usados pelas agências de notícias, jornais e pela polícia. Existe um tipo especial que usa fotografia de prata para gravar informação em papel, contudo ele é de uso bastante restrito e muito pouco deles estão em funcionamento com o tradicional cilindro de varredura e sistema de transmissão analógico.

No tocante a cores, existem duas categorias de FAX à cores, ou seja :

a) Fac-símile de cor completo
b) Fac-símile de muticores

Estes dois tipos de FAX são então descritos sucintamente a seguir.

II.4.1 - FAC-SÍMILE DE COR COMPLETO

É usado para obter imagens de notícias para transmissão posteriormente por TV. No transmissor, a informação de cor da imagem original é separada por um espelho especial para gerar as componentes básicas de cor (vermelha, verde e azul) e através de um chaveamento pode-se obter as amostras de cada cor a serem transmitidas para o aparelho FAX de recepção.

No receptor, usa-se um filme colorido para gravação de informação a "glow tube" onde a luminância é modulada pelos sinais R (vermelho), G (verde) e B (azul) e os raios de modulação são passados em um filtro rotativo que gira numa velocidade equivalente a 1/3 da velocidade do cilindro de transmissão. Assim, gera-se um circuito de pontos que, através do conversor de circuito para linha, são transformados numa linha reta. A informação resultante é então gravada num filme "polaroid".

II.4.2 - FAC-SÍMILE DE MULTICORES

Este tipo de fac-símile produz duas cores, por exemplo, preto e vermelho, e pode ser de 3 cores se considerarmos que o branco é uma cor de fundo. Este tipo de FAX é usado entre as editoras e firmas de impressão de livros. O aparelho desse tipo usa a técnica eletrostática para imprimir os documentos.

No transmissor, as duas fontes de luz para as duas cores, preto e vermelho, são alternadas para obter a separação da informação de cada cor a partir do documento original de duas cores. No receptor, usa-se gravação eletrostática onde uma voltagem positiva imprime os sinais de vermelho e uma voltagem negativa imprime os sinais de preto. Posteriormente, usa-se um "toner" vermelho com polaridade negativa e um "toner" preto com polaridade positiva para a fixação das revelações no papel eletrostático. Após as operações de revelação e fixação de ambas as cores (vermelho e preto), o resultado é uma foto de duas cores.

Capítulo III - Descrição das Ferramentas de Trabalho

III.1 - Linguagem de Máquina

Para que se possa haver uma comunicação entre dois pontos, existem várias maneiras de uma pessoa se expressar as suas idéias que tem em mente, tal que a mensagem possa ser recebida e entendida por outra pessoa. A troca de informação deve ser feita através de uma linguagem comum para o emissor e receptor. Da mesma maneira, o microprocessador deve receber informações, interpretá-las e executá-las, porém o microprocessador e os demais circuitos auxiliares que formam um sistema, usam sinais elétricos para a troca de informações. Assim, a maneira mais prática de codificar as informações é na forma binária.

Um programa a ser executado por um microprocessador deve estar armazenado em sua memória de forma que em cada posição deve existir uma informação codificada em oito dígitos binários. O barramento transmite as informações codificadas na forma binária e as operações lógicas e aritméticas na unidade central de processamento são executadas de acordo com a lógica binária. Assim sendo, as instruções, os dados e os endereços devem ser codificados na forma binária para que eles possam ser processados pelo microprocessador.

Essa maneira de codificar as informações é bastante simples e rápida para uma máquina, porém para um programador é difícil e trabalhosa. A figura III.1(a) ilustra um programa escrito em linguagem de máquina codificada na forma binária, o que corresponde exatamente ao conteúdo da memória caso o programa tenha sido colocado a partir do seu endereço inicial. Este programa ilustrativo está escrito na linguagem que é compreendida pelo microprocessador e, assim, este tipo de linguagem é chamado de linguagem de máquina. Esta linguagem é própria de cada microprocessador e é definida pelo próprio fabricante, podendo ser diferente para cada microprocessador.

ENDERECO	CONTEUDO
0000000000000000	00111110
0000000000000001	10000000
0000000000000010	11010011
0000000000000011	00011111
0000000000000100	00100001
0000000000000101	00000000
0000000000000110	00101011
0000000000000111	11100010
0000000000001000	01101011
0000000000001001	00000101
0000000000001010	11000101
0000000000001011	11010011
0000000000001100	11111110
0000000000001101	11110000
0000000000001110	11111100
0000000000001111	10101010
0000000000010000	01010101
0000000000010001	11100101
0000000000010010	00000001
0000000000010011	00000010
0000000000010100	11000011
0000000000010101	01110110

(a)-Forma Binária

ENDERECO	CONTEUDO
0000	3E
0001	F0
0003	45
0004	47
0005	76
0006	B0
0007	B6
000B	3F
0009	13
000A	23
000B	10
000C	7E
000D	80
000E	D3
000F	AF
0010	00
0011	17
0012	20
0013	30
0014	41
0015	D4

(b)-Forma Hexadecimal

Fig.III.1 - Programa de Linguagem de Máquina

O programa em linguagem de máquina é longo e confuso para o ser humano, porém, eventualmente, poderia ser representado de forma um pouco mais simples, utilizando-se, ao invés da notação binária, a notação hexadecimal. Esta possibilidade deve-se, ao fato de que a cada 4 dígitos binários possuir um correspondente hexadecimal direto, reduzindo o tamanho da representação do programa. A figura III-1(b) mostra o programa escrito em código hexadecimal na forma reduzida. Outros tipos de sistemas de numeração como o decimal ou o octal poderiam ser usados para a representação do programa em linguagem de máquina, sem que isto acarrete em qualquer alteração no programa. A representação no sistema hexadecimal é a mais usada e mais fácil de ser entendida.

O programa em linguagem de máquina pode ser diretamente processado pelo microprocessador, não requerendo nenhuma decodificação ou reorganização. O programa será executado de uma maneira sequencial e direta. A este tipo de programação dá-se o nome de programa objeto. Assim, sendo, todos programas escritos em linguagem de máquina são

programas objeto, pois pasta carregá-los na memória do sistema que estarão prontos para serem executados.

Como o próprio nome diz, esta é uma linguagem muito mais voltada para a máquina de que para ser humano. Evidentemente, os primeiros computadores eram programados em linguagem de máquina. Entretanto, esta linguagem possui uma série de inconvenientes, tais como, programa muito longo, cansativo de ser escrito e de carregar na memória. Além disso, os programas escritos desta maneira são difíceis de serem entendidas e não ilustram as operações que o microprocessador irá executar. Também são bastante susceptíveis à erros, difíceis de serem encontrados e corrigidos. Com a evolução dos microcomputadores, rapidamente apareceram linguagens mais apropriadas que facilitam o trabalho do programador.

III.2 - Linguagem Assembly

A linguagem assembly ou linguagem simbólica foi o primeiro passo na linguagem de programação pois um programa nesta linguagem contém a mesma sequência de instruções que o programa em linguagem de máquina. Porém, certos números são substituídos por símbolos que são mais ilustrativos para o programador. Além disso, cada linha do programa em linguagem assembly possui uma instrução completa e, portanto, sendo necessárias duas ou três posições de memória para cada linha de programa, dependendo do tamanho da instrução. A primeira simbologia utilizada na linguagem assembly é a substituição do código de instrução pelo mneumônico correspondente ao código de operação e os registradores do operando pelas letras identificadoras do registrador. Os mneumônicos de tôdas as ilustrações dos microprocessadores 8088 são apresentados nos capítulos IV e V, onde serão descritos os parâmetros de cada instrução deste trabalho.

Com a substituição dos números por símbolos, os quais com pouco tempo de uso da linguagem assembly tornam-se familiares, já existe uma significativa melhora para a criação e entendimento do programa. Porém, além desta, existem outras particularidades desta linguagem que facilita a tarefa da programação. Estas outras particularidades são o uso de "labels" e comentários do programa. Os "labels" são nomes simbólicos atribuídos às constantes, posições de

memória ou endereços das portas de entradas e de saída, para evitar que se use o valor numérico no programa.

Os comentários são inseridos numa linha do programa em linguagem assembly após um sinal de ponto e vírgula (;) e servem apenas para ilustrar e explicar o funcionamento do programa. Eventualmente uma linha inteira do programa pode ser utilizada para um comentário desde que seja precedida por um ponto e vírgula. Obviamente, os comentários num programa não são obrigatórios porém, eles são essenciais para a compreensão e correções e/ou modificações futuras no programa.

Desta forma, cada linha de um programa em linguagem assembly é constituída basicamente por quatro campos distintos:

(Label), (código de operação), (operando) e (comentário)

Os microprocessadores possuem características próprias a cada um deles com diferente estrutura interna, com diferente conjunto de registradores e também com um conjunto de instruções diferentes, de um para o outro.

Desta forma, a linguagem assembly é uma linguagem característica para cada microprocessador, não sendo possível um programa que foi escrito em linguagem assembly para um microprocessador 8088 ser processado por um microprocessador 6800, ou seja, a linguagem assembly é uma linguagem específica e diferente para cada microprocessador. Além disso, para um mesmo microprocessador também podem existir algumas variações nas características da linguagem assembly. Assim, para uma mesma estrutura básica do microprocessador, não há uma linguagem universal.

A linguagem assembly é uma evolução da programação em em linguagem de máquina. A programação em linguagem de máquina é uma tarefa desgastante, difícil e toma muito tempo do programador. Por outro lado, a linguagem de assembly usa símbolos (mneumônicos) em lugar de números e dígitos binários ou hexadecimais, tornando-a econômica em termos de quantos bytes de memória são requeridas, rápida para "rodar" o programa e, ao mesmo tempo, complicada. Esta complexidade é devido à interação direta com os registradores internos do microcomputador, os quais exigem um entendimento profundo do funcionamento interno do microcomputador. Assim, o programador nesta linguagem normalmente conhece bem a máquina onde êle desenvolve os programas,

utilizando com vantagem a complexidade da linguagem. Note-se que a linguagem assembly é mais usada para hardware do que para software.

Todas as rotinas usadas no programa de codificação e de decodificação neste trabalho foram desenvolvidas em linguagem assembly com acesso direto às memórias do microcomputador usando o programa Debug do MS-DOS, o qual ainda será descrito neste capítulo. Este programa foi e está sendo usado no projeto de codificação de imagens estáticas no Departamento de Comunicações da FEE/UNICAMP.

Obviamente existem outros compiladores em linguagem de alto-nível como a linguagem C, ou seja, a C-51 da Arquimeds para microcontroladores da Intel 8051 e também os compiladores em C para 8088, 80188, 80186, 80286, 80386. Estes compiladores geram códigos para cada CPU que são razoavelmente eficientes e econômicos. Alguns aspectos da linguagem de alto nível são descritos a seguir.

III.3 - Linguagem de Alto Nível

A linguagem assembly já facilitou muito o trabalho de programação. Porém, com o passar dos anos, surgiu a necessidade de por melhores linguagens. Essas novas linguagens são mais voltadas para o problema a ser resolvido, desprendendo-se dos aspectos inerentes ao equipamento que será utilizado e apresentam uma estrutura mais técnica contendo comandos que são decodificadas em diversas instruções das máquinas. Além disso, são linguagens universais que não dependem do repertório de instruções estabelecido pelos fabricantes.

Em linguagem desse tipo, os programas resultantes são mais compactos e eficazes, utilizando mneumônicos que aproximam da formulação do problema sem a necessidade do conhecimento prévio das características da máquina. Além disso, os programas nessas linguagens podem ser transferidos de uma máquina para outra, com pequenas modificações. Assim, o usuário tendo os conhecimentos de sua área específica e dos detalhes da linguagem de alto nível, pode desenvolver os seus programas ou usar os programas prontos. Esses aspectos foram responsáveis pela crescente difusão dos computadores e pela geração de um grande número de pacotes de programas disponíveis.

Nas últimas duas décadas, surgiram centenas de linguagens de alto nível e, atualmente, existem aos milhares, cada uma delas

voltadas para atender uma área de atividade. Existem comissões internacionais que tentam padronizar as linguagens de alto nível para cada área específica de conhecimento para troca de informações pelos computadores do mundo inteiro.

As principais linguagens de alto nível são:

a) BASIC	e) BASICA	i) C	m) ADA
b) BASIEX	f) FORTRAN	j) Turbo C	n) APL
c) Qbasic	g) Cobol	k) C^{++}	o) LOGO
d) Gwbasic	h) Algol	l) PASCAL	

III.4 - Programa de Auxílio Usando BASIEX

Neste projeto foi desenvolvido um programa usando a linguagem BASIEX para inserir os códigos de Huffman na memória do microcomputador. A função deste programa é facilitar a entrada de dados para as duas tabelas de códigos (brancos e pretos) e também para testar e organizar a validade de cada código, antes de armazená-lo em memória.

Há 4 opções neste programa que são:

(a)-Entrar com os códigos
(b)-Salvar os códigos em disquete
(c)-Carregar os códigos em memória
(d)-Encerrar o programa

Na opção de entrada dos códigos, o programa verifica se o comprimento da cadeia de códigos é maior ou igual a zero e menor ou igual a 2048 pontos. Se a cadeia for maior ou um número negativo estiver presente, o programa a rejeita e volta a solicitar uma nova entrada. O programa também detecta se o valor da cadeia é de composição ou de terminação. Dentro da tabela de códigos, o programa oferece opções de se recuar, para mostrar ou modificar o código anterior , avançar para o próximo código, entrar com nova cadeia, voltar para o menu principal do programa e, finalmente, gravar o próprio código de tabela na memória do microcomputador.

Na opção de salvar em disquete, o programa pode salvar os códigos em um disquete para serem carregados no futuro em qualquer

outro microcomputador disponível para trabalho.

Na opção de carregar os códigos na memória, o programa pede o nome de arquivo a ser carregado, define o segmento de trabalho na memória do microcomputador e carrega o arquivo na área definida da memória.

Na opção de encerrar o programa, pergunta-se quer salvar as mudanças. Caso afirmativo, o programa volta para a linguagem BASIEX para salvar as mudanças do programa; caso contrário, o programa pergunta se quer sair do programa. Dependendo da resposta, o programa volta para a instrução inicial ou manda um aviso de saída do programa para o sistema MS-DOS.

Neste programa, há várias subrotinas que ajudam o programa principal a funcionar melhor. Estas subrotinas são descritas nos próximos parágrafos.

ROTINA CONVERTE

A função desta rotina é transformar uma cadeia de dígitos em dois números hexadecimais, X_1 e X_2, sendo X_1 o número menos significativo e X_2 o número mais significativo.

ROTINA VERIFICA

A função desta rotina é garantir que o comprimento do código de pontos brancos ou pretos seja maior do que dois dígitos binários e menor do que 14 dígitos binários, além de verificar que não há qualquer dígito ou símbolo que não seja dígito binário (0 ou 1). O resultado deste teste retorna para o programa principal com a letra "T" para código válido e com a letra "F" para código inválido.

ROTINA ARMAZENA

Esta rotima coloca os dois números X_1 e X_2 na memória do microcomputador, no endereço especificado pela variável ENDE. Esta operação é feita através da instrução POKE do BASIEX. A colocação segue a seguinte forma:

1ª byte ENDE 2ª byte ENDE + 1 3ª byte ENDE + 2 4ª byte ENDE + 3

Número de caracteres nos dois números X_1 e X_2	Ø	X_1	X_2
		número menos significativo	número mais significativo

A rotina coloca no primeiro byte o número de dígitos do código, deixa vazio o segundo byte, armazena no terceiro byte o número menos significativo (X_1), e guarda no quarto byte o número mais significativo (X_2).

III.5 - MICROCOMPUTADOR

A origem do microcomputador teve uma ligação muito forte com a descoberta do microprocessador. Por volta de 1960 os pesquisadores começaram a projetar os circuitos integrados (CI) este envolvia a integração de vários componentes tais como capacitores, transistores e resistores em um circuito integrado singular. O avanço da tecnologia de CI levou à descoberta de microcomputador próximo ao ano 1970 quando a Intel introduziu o primeiro microprocessador o INTEL 8008 de 8 bits, o qual foi denominado de o microprocessador de primeira geração. Em 1974, este microprocessador evoluiu para um microprocessador de segunda geração denominada 8080. O sucesso que este microprocessador fez no mercado levou outras companhias a introduzir outros microprocessadores de uso geral como por exemplo Ø Z80 de Zilog, o 6800 da Motorola e o NCS 800 da National semicondutor.

Em 1978, a Intel introduziu o microprocessador de terceira geração denominado INTEL 8086, que oferecia compatibilidade total com o INTEL 8080 e melhorava o desempenho e a estrutura do projeto de fabricação de microprocessadores. Mais tarde, visando melhorar a compatibilidade com os dispositivos de entrada e de saída, fez-se uma simplificação do projeto do INTEL 8086, da qual surgiu o INTEL 8088 que serviu de base para o nosso trabalho.

Em 1983 a IBM escolheu o microprocessador INTEL 8088 para ser o processador de seu computador pessoal, o IMB PC-XT. Esta escolha incentivou a Intel a desenvolver mais microprocessadores para melhorar o funcionamento do INTEL 8086 e do INTEL 8088 através da criação de um processador número ou co-processador denominado INTEL 8087, o qual foi projetado para fazer cálculos em alta velocidade e

computação cientifica de alta precisão. Criou-se também um segund processador, o INTEL 8089, para facilitar o processamento da operações de entrada e saída como por exemplo para fazer interlaçamento das operações de entrada e de saída.

A próximo versão do INTEL 8086 foi o INTEL 80186 qu teve um quantidade maior de instruções que o INTEL 8086 e o INTE 8088. Depois a Intel lançou o microprocessador 80286 usando como processador principal da IBM PC AT-286. Este último teve uma unidad de gerenciamento da memória embutida no processador. Mais tarde, Intel lançou o microprocessador 80386, utilizado pela IBM como sendo seu processador principal do PC AT-386.

A evolução do projeto do INTEL 8086 continuou com lançamento do INTEL 80486 usado pela IBM como o processador princípa do PC AT-486 resultando num microcomputador com frequência mais alta execução de instruções em menor tempo. Atualmente, a Intel desenvolv um microprocessador mais novo que o da Intel (Pentium) e que será processador principal da IBM para o microcomputador PC-AT (Pentium). cada dia que passa surge um novo processador com maior rapidez d processamento e frequência de relógio do processador mais alto r lugar da anterior que sai da linha de produção.

III.6 - IMPRESSORA

A impressora é um periférico usado por microcomputador para imprimir os dados, sejam de textos ou dados gráficos e desenhos de alta resolução. A impressora usada neste projeto é RIMA XT 180 matricial, compatível com microcomputadores da linha IBM PC-XT ou IBM PC-AT. Esta impressora podem ser ligada ao microcomputador via porta paralela ou serial.

Neste projeto, a impressora é usada para imprimir os dados de FAX recebidos e, com algumas modificações físicas, para efetuar a varredura do documento de FAX a ser enviada. Para isto, um sensor é fixado sobre a cabeça de impressão para varrer o documento de FAX usando os micro-movimentos e micro-espaçamentos de impressão.

A impressora possui 2 partes principais, ou seja, a eletrônica e a mecânica. A parte mecânica é a responsável pela recepção de dados e sinais de controle do mecanismo da máquina. Sob o comando do microprocessador, a impressora lê, interpreta o teclado e e coloca as informações no buffer de impressão (memória temporária) para serem impressas posteriormente. O microprocessador controla também tôda parte mecânica da impressora que é composta pelo carro impressor (cabeça de impressão, fita, cartucho da fita, etc), motor de tração do carro (responsável pelo movimento do carro impressor), motor responsável pelo avanço do papel, sensores que identificam o fim do papel e da margem.

Existe dois modos de operação desta impressora. O primeiro é o modo remoto onde a impressora passa a funcionar sob o comando do microcomputador. Neste modo, a impressora não atende a nenhum comando de seu teclado exceto ao da tecla remoto. O outro é o modo local onde a impressora é comandada através do teclado. Neste estado, ela não está apta para receber os dados pela linha de comunicação. Contudo, caso queira, pode-se passar a este modo, ainda que a impressora esteja imprimindo algum documento.

A estrutura de controle desta impressora consiste de um microcomputador, memória RAM que pode servir como buffer de impressão, dispositivo de entrada e saída para controlar e gerenciar o

teclado de comunicação com o operador, dispositivo de entrada e saída para controlar e gerenciar a parte mecânica da impressora e o dispositivo de interface com o microcomputador (serial ou paralela).

Neste projeto de trabalho, a impressora é usada em modo REMOTO que requer um envio de instrução pelo microcomputador. Qualquer instrução é enviada para a impressora por meio de sinais codificados em código ASCII (American Standard Code Information Interchange). Além dos textos, pode-se também enviar os sinais correspondentes de controle que determinam, por exemplo, o modo gráfico de impressão, o espaçamento entre os pontos horizontais e entre as linhas consecutivas. Neste modo, a precisão de impressão de um ponto pode chegar até (1/216)", tanto no sentido vertical como horizontal.

Uma das formas de enviar as instruções codificadas à impressora é através de um programa em linguagem de alto nível com Basic, Pascal, C, etc. No caso de Basic, as instruções devem ser sempre precedidas de um comando LPRINT (ou PRINT em alguns micros). As instruções que determinam o tipo de impressão serão dadas por meio de comando CHR$(N), às vezes seguido de outros parâmetros.

Outra forma de enviar instruções à impressora é através de um programa em linguagem Assembly, e as instruções que determinam o tipo de sinais de controle serão dadas por meio de comando 1BH.

III.7 - PROGRAMA DE DEPURAÇÃO "DEBUG"

O Debug é um programa de depuração que oferece um ambiente controlado de teste para arquivos objeto binários e executáveis. O Debug tem uma função semelhante ao editor de linhas do M-DOS "Edlin", que é usado para alterar arquivos fontes, como por exemplo o AUTOEXEC.BAT e o CONFIG.SYS. O Debug é usado para alterar arquivos binários executáveis, como COMMAND.COM, DISKCOPY.COM, BACKUP.COM, NLSFUNC.EXEC, DOSKEY.EXE, SET.EXE, SETVER.EXE, ou qualquer outro arquivo com extensão COM ou EXE. O Debug permite também a alteração do conteúdo de um arquivo, de um registro da unidade central de processamento (CPU) ou de uma área de memória do microcomputador e a execução imediata do programa para verificar o resultado das alterações feitas.

O Debug elimina a necessidade de remontar um programa para verificar se um problema foi resolvido por uma pequena alteração. O usuário pode executar o programa logo após a sua edição. O Debug pode ser iniciado de dois modos. No primeiro, deve-se teclar todos os comandos em resposta ao aviso do Debug (hífen "-") e, no segundo, teclar todos os comandos na linha utilizada para iniciar o Debug.

MODO 1 : Debug

MODO 2 : Debug [nome de arquivo [extensão]]

MODO 1 : Debug

Para iniciar o Debug pelo modo 1, tecla-se apenas "Debug", e o Debug responde com o aviso de hífen (-), informando que está pronto para executar os seus comandos. Uma vez que não se especificou um nome de arquivo, podem ser usados outros comandos para trabalhar com a memória atual, setores de disco, ou arquivos.

MODO 2 : Linha de comandos

Neste modo, inicia-se dando o comando:

Debug [nome de arquivo [extensão]]

Por exemplo, se for especificado o nome de um arquivo, o comando é:

Debug arquivo exe

O Debug então carrega o arquivo.exe na memória a partir do endereço 100H, no primeiro segmento disponível. Os registradores BX : AX são carregados com o número de bytes colocados na mémoria.

Caso se inclua um nome de arquivo, pode-se também especificar uma extensão. A extensão é uma lista de parâmetros e opções que é passada para o programa (nome de arquivo). Desta forma, quando o nome de arquivo é carregado na memória, será mais fácil a sua identificação e execução.

Cada comando do Debug é composto de uma única letra, seguida de um ou mais parâmetros. Além disso, há caracteres de controle e funções especiais de edição descritos em qualquer livro de referencia do MS-DOS.

Se um erro de sintaxe ocorrer em um comando do Debug, ele indica o erro com um circunflexo (^) e a palavra "ERRO". Pode-se usar qualquer combinação de letras maiúsculas e minúsculas para

digitar comandos e parâmetros.

O debug é um programa muito poderoso que pode ser utilizado para depurar programas em assembly, mas deve-se tomar muito cuidado com o seu uso, principalmente com o comando de E (entrada de dados). Entrando com os dados numa posição de memória errada ou entrando com dados errados pode causar muitas surpresas. Pode aparecer uma tela cheia de caracteres estranhos, travar o teclado do microcomputador, ou pode levar o DOS a interromper o Debug e precisará carregar de novo de um diskete. Embora não cause danos físicos ao microcomputador, pode-se ter muitos resultados estranhos e inexplicáveis e perder todos os dados digitados durante a sessão de Debug.

Os comandos do Debug estão relacionados abaixo, com uma breve descrição da função de cada comando.

COMANDO DO DEBUG	FUNÇÃO
A [endereço]	Assemble (Este comando aciona o começo da amostragem de um programa em Assembly)
C faixa endereço	Compare (Este comando faz uma comparação entre dois blocos de instruções ou dois blocos de memória)
D faixa	Dump (Exibe o conteúdo da faixa de memória espeficada na tela)
E endereço lista	Enter (Insere os valores de bytes no endereço especificado)
F faixa lista	Fill (Preenche os endereços da faixa especificada com os valores contidos na lista especificada)
G [= endereço]	Go (Executa o programa atualmente na memória começando no endereço especificado)
H valor	Hex (Executa a soma e subtração dos dois valores especificados, em Hex) . Primeiro exibe a soma, e depois exibe a subtração.
I valor	Input (Lê e exibe o byte da porta de entrada especificada pelo valor)

COMANDO DO DEBUG	FUNÇÃO
L [endereço [unidade:registro]]	Load (Carrega um arquivo na memória começando no endereço especificado)
M faixa endereço	Move (Move o bloco de memória especificado pela faixa para a posição que inicia no endereço especificado)
N nome de arquivo	Name (Define nomes de arquivos)
O valor byte	Output (Envia o byte especificado para a porta de saída especificada pelo valor)
R [nome do registrador]	Register (Exibe o conteúdo de um ou mais registradores da CPU. Pode chegar no máximo até 15 15 registradores).
S faixa lista	Search (Pesquisa na faixa especificada, a lista especificada de bytes)
T [= endereço] [valor]	Trace (Executa uma instrução e exibe na tela o conteúdo de todos os registradores, "flags" e a instrução decodificada)
U [faixa]	Unassemble ("Desmonta" bytes e mostra na tela as intruções de origem correspondentes aos mesmos, com endereços e valores de byte)
W [endereço [unidade:registro]]	Write (Grava em disco rígido ou flexível o arquivo que está sendo depurado)
Q	Quit (Encerra operação do programa debug)

III.8 - HARDWARE DO PC-XT

A unidade do sistema de PC-XT utilizado contém uma placa de sistema que consiste de 8 status de expansão, de um microprocessador 8088, de 40 Kbytes de memória ROM, de 128 Kbytes de memória RAM, de um circuito de alto-falante, de dois canais de entrada e saída e três temporizadores/contadores programáveis.

As memórias ROM ("Read Only memory") somente permitem ler as informações nelas gravadas. Nessas memórias estão o programa básico do sistema de entrada e saída (Bios), o programa de inicialização do microcomputador e todas as rotinas iniciais de programa na hora que liga o microcomputador.

As memórias RAM ("Random Access Memory") permitem a leitura e escrita de forma aleatória à qualquer endereço. A memória RAM, na verdade, é uma folha de rascunho ou uma folha de trabalho onde se escreve e/ou apaga as informações que variam de um momento para outro.

O primeiro temporizador/contador é de uso geral como, por exemplo, gerar a base de tempo para relógio de tempo real, o segundo temporizador/ contador é usado para a geração de TONS para circuito de alto-falante e o terceiro é usado para a geração de tempo para acesso direto à memória de dados. Todos os temporizadores tem uma resolução mínima de 1.05 ms.

A placa de sistema também contém oito níveis de prioridade de interrupção controlados por software, dos quais seis são usados pelas placas de expansão e dois são usados pela placa de sistema. O nível Ø tem a prioridade mais alta e conectado com canal Ø do temporizador/contador para fornecer uma interrupção periódica para o relógio de tempo real. O nível 1 é conectado com circuito de interface de teclado e recebe uma interupção para cada código de varredura mandado pelo teclado.

A placa de sistema também dá apoio às memórias ROM e RAM com um espaço para 64 Kbytes de ROM ou EPROM. Dois módulos são fornecidos,onde cada módulo aceita dispositivo de 32 Kbytes ou 8 Kbytes. Um tem 32 Kbytes de ROM e outro tem 8 Kbytes. Este ROM contém o programa de auto teste na hora que liga o microcomputador, as rotinas dos canais de entrada e saída com padrões para 128 caracteres no modo gráfico e também as rotinas de aceitação e reinicialização do microcomputador através de um diskete.

III.9 - Estrutura Interna do Microcomputador

Os registradores de uso geral tem que ser fornecidos para que a unidade aritmética lógica (ALU) possa manipular os dados numa velocidade alta por causa das restrições no número de bits em cada instrução e das limitações no número de registradores que podem ser endereçados diretamente para cada microprocessador. Por exemplo, para 8080 são 8 registradores que podem ser endereçados diretamente enquanto que para 8088 são 16 registradores.

Cada registrador é formado por oito flip-flops no caso do registrador de 8 bits e 16 flip-flops no caso do registrador de 16 bits. Estes registradores são conectados a um barramento interno bi-direcional de dados, permitindo uma transferência de dados do ou para o barramento.

A implementação deste registrador com flip-flops de tecnologia MOS fornece um nível mais rápido de memória disponível, e o conteúdo de cada registrador pode ser acessado em poucos nanosegundos. A tarefa de cada registrador de uso geral não é definida à priori e, como são de uso geral, podem conter qualquer dado usado por um programa aplicativo. Estes registradores podem ser usados como um registrador singular de 8 bits ou juntar dois registradores de 8 bits para formar um registrador de 16 bits.

III.10 - Definições

CPU

é a Unidade Central de Processamento que contém duas partes. A primeira é a unidade de operações lógicas e aritméticas, e a segunda é a unidade de controle. A função da primeira é efetuar as operações aritmeticas e lógicas com os dados que passam por esta unidade. As funções típicas de aritmética podem incluir a soma, e a subtração, e as funções típicas de lógica podem incluir porta AND, porta OR e operações de deslocamento.

A responsabilidade da unidade de controle é sequenciar as operações do sistema todo; em particular, vai gerar e administrar

todos os sinais de controle necessários para sincronizar as operações e fluxo de dados dentro e fora da unidade de operações lógicas e aritméticas (ALU). Esta unidade controla também o fluxo de informações pelo barramento de dados, barramento de endereço e barramento de controle.

MICROPROGRAMA

Um programa interno não é acessível ao programador, o qual controla as operações e funções da unidade de controle. Este programa interpreta o conjunto das instruções externas.

CONJUNTO DE INSTRUÇÕES

É uma lista de todas as instruções disponíveis para o programador poder atribuir as funções para o microprocessador. Por exemplo, atribuir a função de somar o conteúdo do registrador 1 com o conteúdo do registrador 2.

PROGRAMA

É uma seqüência de instruções que foram escritas por um usuário. Essas instruções são codificadas em sistema binário para serem inseridas na memória do microcomputador. Cada uma das instruções consecutivas são lidas sob a supervisão da unidade de controle e serão depositados num registro especial da unidade de controle onde serão decodificados e executados.

BARRAMENTO DE DADOS

Este barramento leva os dados entre as unidades do sistema. Um microprocessador de 8 bit precisa de um barramento de dados de 8 bits. O barramento de dados é bidirecional, ou seja, ele pode levar ou trazer os dados nos dois sentidos.

BARRAMENTO DE ENDEREÇO

Este barramento é usado para selecionar a origem e o destino dos sinais transmitidos por outros barramentos. Tipicamente, este barramento é usado para selecionar um registrador nas várias unidades do sistema que pode ser usado como a fonte ou o destino de dados. Normalmente, o barramento de endereço tem 16 linhas que pode endereçar até 64 Kbytes de memória.

BARRAMENTO DE CONTROLE

Este barramento é usado para a sincronização do sistema de microprocessador. As linhas deste barramento carregam informações de estado e de controle para a Unidade de Processamento Central. Este barramento precisa ter, no mínimo, 10 linhas diferentes de sinais de controle para ser eficiente.

ENDEREÇO:

O endereço pode ser uma informação de 8 bits correspondente ao endereço de uma posição de memória ou uma informação de 8 bits correspondente ao endereço de uma porta de entrada ou saída. Esta porta pode ser de um contador ou de um outro circuito periférico qualquer do sistema.

PILHA

Uma porção da área de memória reservada para os dados e endereços de retorno, no caso de chamada de sub-rotinas dentro de um programa principal.

APONTADOR DE PILHA (sp)

é um registrador de 16 bits que endereça à pilha. Antes de começar o programa principal, o apontador de pilha deve ser carregado por um endereço inicial de pilha.

Capítulo IV - Codificação

IV.1 - Introdução

As vantagens de um código onde os símbolos das mensagens são de comprimento variável são de que, muitas vezes, ela é mais eficiente no sentido de representar uma mesma informação com o mesmo símbolo, e de poder usar, em média, menos dígitos para cada símbolo. Para fazer isto, é preciso saber algumas informações sobre a estatística da mensagem transmitida. Se cada símbolo tem a mesma probabilidade de ser enviado do que qualquer outro símbolo, então o código de bloco é tão eficiente quanto qualquer outro código, mas se algums símbolos são mais prováveis de serem transmitidos do que os outros símbolos, então pode-se aproveitar deste fato e usar um código curto para representar os símbolos mais freqüentes e usar códigos mais compridos para os símbolos menos freqüentes.

É exatamente isto que Código de Morse faz. A letra E na língua inglesa é usada com maior frequência do que as outras letras. Então, Morse lhe atribuiu um código mais curto que é um ponto (●) e atribuiu códigos mais compridos às letras menos usadas na língua inglesa.

Neste projeto, usa-se o código de Huffman modificado (CHM). Este código usa a mesma idéia do código de Morse. Foram estudadas várias mensagens diferentes tais como cartas comerciais, manuscritos, mapas meteorológicos, padrões de teste e desenho de circuitos eletrônicos. O resultado desses estudos foi analisado do ponto de vista quantitativo e qualitativo e também foi analisado do ponto de vista estatístico da informação de cada mensagem. Esta estatística mostrou que as correntes de pontos brancos e pretos têm uma relação inversa com a probabilidade de ocorrência de cada corrente de pontos. Portanto, às correntes de pontos que têm uma probabilidade de ocorrência maior foram atribuídas um código de comprimento menor ou curto e àquelas correntes de pontos que têm uma probabilidade de ocorrência menor foram atribuídas um código de comprimento maior ou longo. Assim, em média, pode-se representar mais informações para cada símbolo de código.

No código de Huffman Modificado, foi dado o código "11"

para uma corrente de 2 pontos pretos e o código "10" para uma corrente de 3 pontos pretos. Estas duas correntes são as mais freqüentes nas mensagens estudadas e, por isto, receberam os códigos mais curtos. Por outro lado, foi dado o código mais longo (00000011011) à corrente de 512 pontos pretos e um código de igual tamanho para o controle de 576, 640, 704, 832, 896 pontos pretos porque estas correntes ocorrem com probabilidades menores.

Neste projeto, usa-se o número de pontos pretos ou brancos em cada corrente como um índice para endereçar à tabela de códigos. Então, o próprio número de pontos serve como endereço para localizar o seu código.

IV.2 - Descrição Geral do Processo de Codificação

Os dados com a informação gráfica são captados pelo dispositivo de carga acoplada ou sensor óptico (CCD = Charged Coupled Device). Este dispositivo transforma a informação óptica em um trem de pulsos digitais que podem representar dois ou mais níveis de luminância, dependendo de número de níveis de graduação de cinza selecionado.

O trem de pulsos provenientes do "CCD" é chamado de linha original de informação, a qual é gravada na memória do microcomputador em uma área que começa a partir do endereço 3000:0000. Após colocar os dados nesta área, pode-se começar o processo de codificação dos dados da imagem de fac-símile.

O processo começa contando os pulsos de níveis zero e um, onde um "zero" (0) significa um ponto branco, e "um" (1) significa um ponto preto. Portanto, a rotina conta quantos zero's existem em cada corrente de zero e quantos um's existem em cada corrente de um. Deve-se realizar uma varredura bastante precisa sobre uma página de informação manuscrita ou digitada tal que a varredura de uma linha horizontal, cuja espessura é da ordem de 0,25 mm, detecte as correntes de pontos brancos e pretos.

Quando o processo da contagem de pontos termina, pode-se consultar a tabela dos códigos correspondentes. Neste caso, usa-se um código de Huffman Modificado, conforme mostrado na figura IV.1. . Cada corrente de pontos brancos ou pretos tem um código. Em seguida , processa-se uma linha inteira , a qual é chamada de linha codificada que pode abranger mais do que uma linha original. As linhas codificadas são então agrupadas para formar um buffer de transmissão contendo todos os códigos de todas as correntes de pontos brancos ou pretos.

Há várias rotinas auxiliares que ajudam a melhorar o funcionamento deste programa. Por exemplo, existe a rotina "compose" que divide as correntes de pontos pretos ou brancos em múltiplos de 64 pontos, formando as correntes de composição e o restante das correntes que tem menos do que 64 pontos formam as correntes de terminação.

Fig.IV.1 - Tabela de Códigos para as Correntes de Pontos Pretos e Brancos

CORRENTE BRANCA	PALAVRA CÓDIGO	CORRENTE PRETA	PALAVRA CÓDIGO
0	00110101	0	0000110111
1	000111	1	010
2	0111	2	11
3	1000	3	10
4	1011	4	011
5	1100	5	0011
6	1110	6	0010
7	1111	7	00011
8	10011	8	000101
9	10100	9	000100
10	00111	10	0000100
11	01000	11	0000101
12	001000	12	0000111
13	000011	13	00000100
14	110100	14	00000111
15	110101	15	000011000
16	101010	16	0000010111
17	101011	17	0000011000
18	0100111	18	0000001000
19	0001100	19	0000011001111
20	0001000	20	0000011010000
21	0010111	21	0000011011100
22	0000011	22	0000010110111
23	0000100	23	0000001010000
24	0101000	24	0000000010111
25	0101011	25	0000000011000
26	0010011	26	0000011001010
27	0100100	27	0000011001011
28	0011000	28	0000011001100
29	00000010	29	0000011001101
30	00000011	30	0000001101000
31	00011010	31	0000001101001
32	00011011	32	0000001101010
33	00010010	33	0000001101011
34	00010011	34	0000011010010
35	00010100	35	0000011010011
36	00010101	36	0000011010100
37	00010110	37	0000011010101
38	00010111	38	0000011010110
39	00101000	39	0000011010111
40	00101001	40	0000001101100
41	00101010	41	0000001101101
42	00101011	42	0000011011010
43	00101100	43	0000011011011
44	00101101	44	0000010010100
45	00000100	45	0000010010101
46	00000101	46	0000010010110
47	00001010	47	0000010010111
48	00001011	48	0000001100100
49	01010010	49	0000001100101
50	01010011	50	0000010100010
51	01010100	51	0000010100011
52	01010101	52	0000000100100
53	00100100	53	0000000110111
54	00100101	54	0000000111000
55	01011000	55	0000000100111
56	01011001	56	0000000101000
57	01011010	57	0000010011000
58	01011011	58	0000010011001
59	01001010	59	0000000101011
60	01001011	60	0000000101100
61	00110010	61	0000010011010
62	00110011	62	0000001100110
63	00110100	63	0000001100111

CORRENTE BRANCA	PALAVRA CÓDIGO	CORRENTE PRETA	PALAVRA CÓDIGO
64	11011	64	0000001111
128	10010	128	000011001000
192	010111	192	000011001001
256	0110111	256	000001011011
320	00110110	320	000000110011
384	00110111	384	000000110011
448	01100100	448	000000110101
512	01100101	512	0000001101100
576	01101000	576	0000001101101
640	01100111	640	0000001001010
704	011001100	704	0000001001011
768	011001101	768	0000001001100
832	011010010	832	0000001001101
896	011010011	896	0000001110010
960	011010101	960	0000001110011
1024	011010101	1024	0000001110100
1088	011010110	1088	0000001110101
1152	011010111	1152	0000001110110
1216	011011000	1216	0000001110111
1280	011011001	1280	0000001010010
1344	011011010	1344	0000001010011
1408	011011011	1408	0000001010100
1472	010011000	1472	0000001010101
1536	010011001	1536	0000001011010
1600	010011010	1600	0000001011011
1664	011000	1664	0000001100100
1728	010011011	1728	0000001100101
EOL	000000000001	EOL	000000000001

Então, o código é composto da seguinte maneira:

$$\text{Palavra Código Final} = 64m + n$$

onde $m = 0, 1, 2, \ldots, 32$ e $n = 0, 1, 2, \ldots, 63$. Por exemplo, para uma corrente de 92 pontos brancos ($92 = 64 + 28$), a palavra código final é formado por um código de composição de 64 pontos brancos e um código de terminação de 28 pontos brancos.

Neste caso, vale notar também que a seqüência de sincronização de "fim de linha" (FDL) é uma seqüência única e não pode ser produzida por nenhuma combinação de palavras-códigos, porque todas as palavras códigos começam, no máximo, com 6 zeros enquanto que a FDL tem 11 zeros.

O resultado final do programa de codificação é um buffer de transmissão que contém todas as palavras-códigos de uma página inteira de informação. Estes códigos são organizados de forma compacta e adequada para a transmissão bit a bit, permitindo que uma página inteira seja transmitida em um tempo mínimo (normalmente 1 minuto, no caso de usar-se uma máquina de fac-símile do grupo 3).

IV.3 - Rotina de Inicialização

```
XOR  AX,  AX      ;    Zera Acumulador
XOR  BX,  BX      ;    Zera o Registrador BX
XOR  CX,  CX      ;    Zera o Registrador CX
XOR  DX,  DX
XOR  DI,  DI
XOR  BP,  BP
XOR  SI,  SI      ;    Zera o Registrador SI
PUSH DS           ;    Salva DS na Pilha
NOP
MOV  AX,  3000    ;    Define segmento de dados
MOV  DS, AX
NOP
XOR  AX,  AX
MOV [FFFE],AX     ;    Zera o contador do número de linhas já
                       processadas
MOV  DI,  FFFE
DEC  DI
DEC  DI
MOV [DI], AX      ;    Zera o ponteiro de endereço da última
SUB  DI,  Ø4h     ;    Palavra usada em BUFTX
MOV [DI], AX      ;    Zera o contador dos bits usados
INC  DI           ;    Na última posição de BUFTX
INC  DI
MOV  AX,  4000    ;    Inicializa ponteiro do segmento de
                       trabalho atual
XOR  DI,  DI
MOV  BX,  10h     ;    Começa o batente com valor inicial
                  ;    10h
```

Em suma, esta rotina tem a função de sincronizar as operações do programa de codificação colocando os valores iniciais em cada registrador e posição de memória para evitar os erros e problemas na hora de executar o programa de codificação. A figura IV.2 mostra um resumo da organização da memória do programa de codificação.

Fig.IV.2 - Organização da Memória do Programa de Codificação

SEGMENTO OFFSET	EXPLICAÇÃO "CONTEÚDO	TAMANHO	ESTADO INICIAL
3 000 : 0 000	Área onde se guardam os bits de informações grafi- cas da linha original	256 bytes por linha	vazio
3 000 : 7 C00	Endereço de códigos de cor rentes de pontos brancos	384 a 512 bytes	codigo de pontos
3 000 : 7 C00	Endereço de códigos de cor rentes de pontos pretos	384 a 512 bytes	codigo de pontos
3 000 : 8 000	Buffer do codigos "Buf Cod"		vazio
3 000 : F FF6	Ponteiro da primeira posi ção vazia do "Buf Cod"	palavra	vazio
3 000 : F FF8	Contador dos bits usados na na última posição de BUFTX	palavra	vazio
3 000 : F FFA	Um ponteiro que índica o segmento atual	palavra	3 000
3 000 : F FFC	Endereço da última palavra usada em BUF TX	palavra	4 000 : 0 000
3 000 : F FFE	Contador de número de li nhas já processadas	palavra	vazio
4 000 : 0 000	Inicio do Buffer de Trans missão "BUF TX"		vazio

IV.4 - ROTINA T

```
T :  MOV  DI,  8000H
     XOR  SI,  SI
     CALL CCD
     JMP  X
```

A função básica desta rotina é colocar o endereço inicial do buffer de codificação no registrador DI. Este buffer é uma área da memória onde são armazenados os códigos de pontos pretos e brancos. Além disso, internamente, esta rotina chama a rotina CCD para colocar as linhas originais de informações gráficas.

A rotina CCD simula as informações gráficas ou um trem de pulsos que vêm de um sensor óptico, visto que o hardware do circuito de aquisição de dados não foi construído. A rotina CCD coloca os dados na área de memória do computador genericamente, para simular uma ou várias linhas originais de informação gráfica (pontos pretos e brancos). Ela é apresentada a seguir.

IV.5 - ROTINA CDD

```
CDD: MOV  AX,  3000H
     MOV  DS,  AX
     XOR  CX,  CX
     XOR  SI,  SI

VOLTA: MOV AX, FFFF
     MOV [SC], AX
     INC  SI
     INC  CX
     XOR AX, AX
     MOV [SI], AX
     INC  SI
     INC  CX
     CMP  CX,  Valor = 100
     JNE  VOLTA
     XOR  SI,  SI
     XOR  CX,  CX
     XOR  AX,  AX
     RET
```

Basicamente, esta rotina coloca as informações gráficas na área de memória que começa no endereço 3000H, para simular as correntes de pontos pretos e brancos recebidas do sensor óptico (CCD).

IV.7 - Rotina Preto e Branco

```
PRETO: MOV  BP,  7E00
VOLTAP: INC  CX            ;  Conta os pontos pretos
      Y: CMP  CX,  BX
         JE   A            ;  Acabou os 16 pontos pretos
                           ;  Trazer mais 16 pontos
         SHR  AX,  1
         JNC  B            ;  Quebrou corrente de pontos pretos
         JMP  VOLTAP       ;  Não quebrou e corrente,  portanto
                              continua
                           ;  Contando
BRANCO: MOV  BP,  7C00
VOLTAB: INC  CX            ;  Conta os pontos brancos
     Z: CMP  CX,  BX       ;  Acabou os 16 pontos brancos
                           ;  Traz mais 16 pontos
         JE   A
         SHR  AX,  1
         JC   B            ;  Quebrou a corrente de pontos brancos
         JMP  VOLTAB       ;  Não quebrou a corrente, portanto
                              continua
                           ;  Contando
```

Rotina de Pretos

Esta rotina examina a corrente de pontos pretos, bit a bit, contando os pontos pretos em cada corrente de pontos pretos. No final de 16 pontos, caso continue preto, a rotina solicita para trazer mais 16 pontos. O processo de contagem assim continua até que a corrente de pontos pretos seja quebrada. Neste caso, o controle é transferido para a rotina "B" que muda a rotina de processamento de pontos brancos.

Rotina de Brancos

Esta rotina examina a corrente de pontos brancos, bit a bit, contando quantos pontos brancos tem em cada corrente de pontos brancos. No final de 16 pontos, caso continue branco, a rotina solicita para trazer mais 16 pontos através da rotina "A". A rotina continua o processo de contagem até que a corrente de pontos brancos seja quebrada. Neste caso, o controle é transferido para a rotina "B" que muda para a rotina de processamento de pontos pretos.

IV.8 - Rotina B (B = Trans)

```
B:   PUSH AX
     PUSH SI
     PUSH DS
     NOP
     PUSH CX
     MOV  AX,  3000H
     MOV  DS,  AX
     NOP
     MOV  SI,  BP
     CMP  CX,  40H
     JAE  COMPOS
     ADD  CX,  CX
     ADD  CX,  CX
     ADD  SI,  CX
     CALL      RESOLVE
     JMP  R1
```

A função desta rotina é analisar as correntes de pontos pretos ou brancos para definir se a corrente é de terminação (até 63 pontos) ou de composição (64 pontos ou mais). Caso os pontos sejam da corrente de composição, a rotina transfere o controle para a rotina "COMPOS" para o processamento dos pontos. Caso os pontos sejam da corrente de terminação, a rotina B determina o endereço de consulta da tabela de códigos para, em seguida, chamar a rotina "RESOLVE" que procura o código que pertence ao número dos pontos encontrados. Depois a rotina B transfere o controle para a rotina R_1.

IV.9 - ROTINA R_1

```
TERM: R1: CMP  DX,  810H
          JE   FDL
          JMP  R2
```

Esta rotina testa se já atingiu o fim do processamento de uma linha original de informação gráfica que contém 2048 pontos de ambas as cores (preto e branco). Atingindo o fim de uma linha original, o controle passa para a rotina FDL. Caso contrário, a rotina segue para a rotina R_2 .

IV.10 - ROTINA R_2 (OFFSET)

```
Rz:  POP  CX
     SUB  BX,  CX
     MOV  CX,  0001
     POP  DS
     MOP
     POP  SI
     POP  AX
     XOR  BP,  1000H
     JZ   Z          ;    BRANCO
     JMP  Y          ;    PRETO
```

Esta rotina, em sequência à rotina R_1, ajusta o conteúdo dos registradores CX e BX para o restante dos bits da linha original e define se a corrente é de pontos pretos ou brancos. Após isto, o controle retorna para a rotina de processamento de pontos pretos ou brancos, respectivamente.

IV.11 - Rotina FDL

```
FDL: MOV  CL,   ØC
     MOV  AX,   0800H
     CALL BUFCOD
     INC  WORD  PTR [FFFE]

     CALL EMENDA
     CMP  WORD  PTR [FFFE], Ø479H
     JB   NFIM
     JE   FIM
```

Esta rotina começa colocando o valor ØCH no registrador CL e o sinal FDL no registrador AX para sincronizar a impressão de informação na recepção. A rotina BUFCOD armazena o sinal FDL no buffer de codificação. Após incrementar o contador de linhas já processadas, é chamada a rotina EMENDA que tem a função de encaixar todos os códigos numa forma compacta e adequada para a transmissão. No final desta rotina, é checado o fim do documento. Em caso afirmativo, o controle passa para a rotina FIM para terminar o programa ; caso contrário, segue-se para a rotina NFIM que irá continuar o processo normal de processamento da informação gráfica.

IV.12 - Rotina A

```
A:  ADD  DX,   10H  ;   Rotina para pegar mais 16 bits
    CMP  DX,   810H ;   De informação ou Elementos de Imagem
    JE   B          ;   De um linha original de informação
    MOV  AX    [SI]
    INC  SI
    INC  SI
    ADD  BX,   10H
    CMP  BP,   0000h
    JE   Z          ;   BRANCO
    JMP  Y          ;   PRETO
```

A função desta rotina é pegar mais 16 bits da linha original, testar se chegou ao fim de uma linha original e verifica a posição correta no processo de codificação. Desta comparação, o controle passa para a rotina que processa os pontos brancos ou para a rotina que processa os pontos pretos.

IV.13 - Rotina Resolve

```
RESOLVE:  MOV  CL,   [SI]
          INC  SI
          INC  SI
          MOV  AX,   [SI]
          CALL BUFCOD
          RET
```

A função básica desta rotina é consultar as tabelas de códigos de Huffman para associar um código de Huffman (normalmente chamado cabeçalho) ao número de bits. Após transferir o código próprio de Huffman para o acumulador, chama a rotina BUFCOD e volta para o programa principal.

IV.14 - Rotina BUFCOD

```
BUFCOD:   MOV [DI],  CL
          INC  DI
          INC  DI
          MOV [DI],  AX
          INC  DI
          IND  DI
          RET
```

A função desta rotina é armazenar mais um ítem na tabela sem compressão ou no buffer de códigos. Inicialmente, ela coloca um cabeçalho na posição do buffer de códigos e depois coloca o próprio código de Huffman na próxima posição vazia do buffer de códigos. Finalmente, ela aponta para a próxima posição vazia do buffer de códigos e volta para a rotina RESOLVE ou para a rotina de onde foi chamada. Nota-se que, nesta rotina, o cabeçalho registra quantos bits há em cada palavra-código. Isto é muito importante para montarmos o buffer de transmissão pois, para isso, é preciso saber exatamente quantos bits há em cada código para encaixá-los de forma compacta para transmissão.

IV.15 - Rotina Emenda

```
EMENDA: MOV  SI,   8000H
        MOV  DI,   [FFFC]
        MOV  CX,   [FFF8]
        MOV  BP,   [DI]
PROX:   MOV  DX,   [SI]
        INC  SI
        INC  SI
        MOV  AX,   [SI]
        INC  SI
        INC  SI
        ROL  AX,   CL
        MOV  BX,   FFFF
        SHL  BX,   CL
        ADD  CX,   DX
        PUSH AX
        ADD  AX,   BX
        XOR  BX,   FFFF
        MOV  DX,   BP
        ADD  DX,   BX
        CMP  CL,   10H
        JAE  PM
        MOV  BP,   AX
        POP  AX
        JMP  FCOD
```

A rotina EMENDA tem como função encaixar todos os códigos numa forma compactada e adequada para transmissão de informação gráfica codificada através de qualquer meio de transmissão. Esta rotina define vários parâmetros tais como endereço do início do BUFCOD, o endereço da última palavra usada no BUFTX (buffer de transmissão) e quantos bits já foram usados na última palavra do BUFTX. Em síntese, ela transfere os códigos de um buffer de código sem compressão para um buffer de transmissão com compressão.

IV.16 - ROTINA FCOD

```
FCOD: MOV  AX,   [FFF6]
      MOV  DX,   SI
      CMP  DX,   AX
      JNE  PROX
      MOV  AX,   BP
      CALL ARM
      MOV  DX,   DI
      MOV [FFFC], DX
      MOV [FFF8], CX
      RET
```

Esta rotina que trabalha juntamente com a rotina EMENDA tem a função básica de verificar se o endereço da primeira posição vazia do buffer de codificação coincide com o endereço do buffer de codificação atual. Caso isto seja verdade, ela chama a rotina ARM para armazenar o código próprio no buffer de transmissão (BUFTX), senão ela chama a rotina PROX para ajustar os dois endereços. Depois de armazenar o código no BUFTX, essa rotina atualiza o ponteiro do endereço da última palavra usada no BUFTX e o ponteiro do contador de quantos bits bits já foram usados para a última posição do BUFTX.

IV.17 - ROTINA PM

```
PM:  PUSH DS
     NOP
     MOV  DX,   [FFFA]
     MOV  DS,   DX
     NOP
     MOV  [DI], AX
     POP  DS
     NOP
     INC  DI
     INC  DI
     JZ   PS
     RET  ?
```

Esta rotina tem a função de armazenar os códigos no BUFTX, atualizar o ponteiro do segmento atual (FFFA) e depois armazenar a palavra-código no BUFTX. Em seguida, leva o ponteiro para a próxima posição vazia no BUFTX e, quando chega ao fim do segmento atual, o controle passa para a rotina PS que transfere o BUFTX para um disco rígido ou flexível.

IV.18 - ROTINA COMPOS

```
COMPOS:  MOV  AX,   CX    ;    Rotina que processa os dados
         DIV  AX,   40H   ;    De composição
         MOV  CX,   100H
         SHL  AL          ;    Desloca quociente duas
         SHL  AL          ;    Posições para esquerda
         PUSH AX
         AND  AX,   00FF  ;    Mascara os dados de
                          ;    Composição
         ADD  CX,   AX
         PUSH SI
         ADD  SI,   CX
         CALL RESOLVE     ;    Chama rotina de consultar
         POP  SI          ;    Tabela de composição
         POP  AX
         SHL  AH          ;    Desloca o resto da divisão
         SHL  AH          ;    Duas posições para esquerda
         XCHG AH,   AL
         AND  AX,   00FF  ;    Mascara os dados de
                          ;    Terminação
         ADD  SI,   AX
         CALL RESOLVE     ;    Chama rotina de consultar a
                          ;    Tabela de terminação
         JMP  R1
```

A função da rotina COMPOS é processar os dados de composição. Para determinar o valor de composição, divide-se o número de pontos por 40 Hex (64 decimal), da qual se pega apenas os dados de composição. Após ajustar o endereço de consulta, chama-se a rotina RESOLVE para consultar a tabela de códigos de composição. O resto da

divisão forma o número de pontos de terminação. Portanto, após ajustar o endereço de consulta, chama-se novamente a rotina RESOLVE para consultar a tabela de código de terminação.

CAPÍTULO V - DECODIFICAÇÃO

V.1 - DESCRIÇÃO GERAL DAS ROTINAS DE DECODIFICAÇÃO

Os dados recebidos são armazenados no buffer de recepção que ocupa uma área de memória do microcomputador começando no endereço 4000:0000h. Conforme mencionado no capítulo anterior, as informações recebidas são as palavras-código que representam os números de pontos pretos e brancos nas linhas originais de informação.

Cada página de informação codificada começa com um caracter FDL, uma palavra-código de 11 "0" e um "1" (000000000001), e termina com uma palavra-código indicando o fim da página que é uma sequência de seis caracteres FDL. Portanto, uma página binária codificada apresenta a configuração mostrada na Fig. V.1.

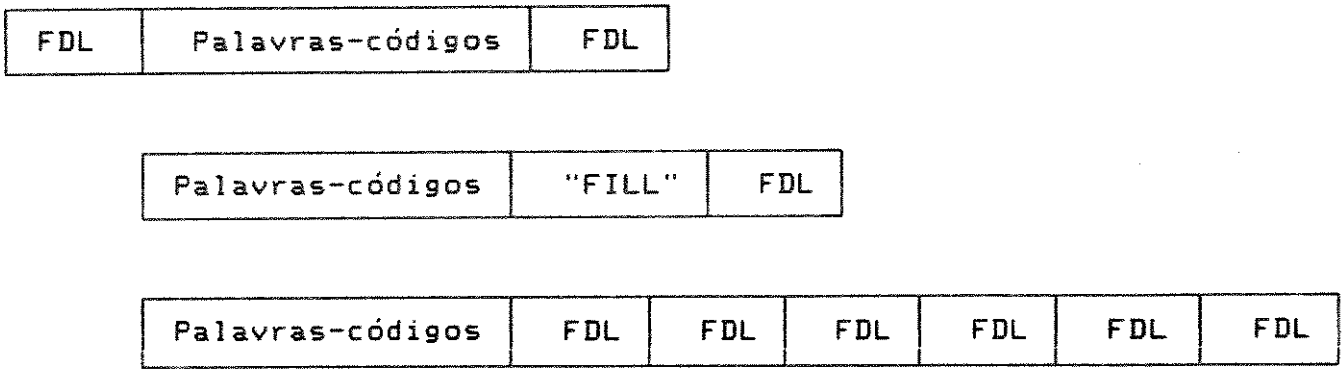


Fig.V.1 - Estrutura de um arquivo FAX codificado

Cada linha codificada tem comprimento variável pois depende da quantidade de informação da linha original. Inicialmente, as informações codificadas são lidas do buffer de recepção para a análise das palavras-códigos, uma por uma. No processo de decodificação há vários inconvenientes, um dos quais é a sincronização devido à natureza de comprimento variável das palavras-códigos (codificação Huffman), as quais são recebidas em sequência, uma após a outra. Em virtude dessa organização, a decodificação torna-se uma tarefa complexa.

Pela Fig. V.1, nota-se que o primeiro código recebido é uma palavra-código FDL. Assim, a primeira providência a ser tomada é

detectar a sua existência. A análise é feita sempre em formato de 16 bits. Como FDL é composto de 11 "0" e um "1", na hora de leitura dos dados do buffer de recepção, normalmente considera-se os primeiros 16 bits contendo um caracter FDL (12 bits) e os quatro bits restantes que podem ser ou não uma palavra-código de uma corrente de pontos brancos ou pretos de um certo comprimento. Por outro lado, se não for detectado um código FDL, então houve algum problema sério com a transmissão e é preciso mandar algum aviso ou mensagem à tela do computador alertando o operador de recepção.

Depois de detectado o código FDL, o processo de decodificação continua, gerando duas tabelas de códigos na memória do microcomputador, uma para os elementos brancos e outra para os elementos pretos. As duas tabelas ocupam um espaço de memória de quase 16 Kbytes (8 Kbytes para pretos, e 8 Kbytes para brancos) e são arranjadas de tal forma que a própria palavra-código recebida sirva como endereço para encontrar o número de pontos pretos ou brancos representados pelo código. Esta técnica facilita bastante o processo de decodificação pois, quando uma palavra-código é recebida, ela vai para o registrador de endereços onde se verifica quantos pontos brancos ou pretos o código carrega. A formação das tabelas de composição e de terminação é mostrada nas Figs. V.2 e V.3, respectivamente.

PALAVRA CÓDIGO	CORRENTE BRANCA	PALAVRA CÓDIGO	CORRENTE PRETA
11011	64	0000001111	64
10010	128	000011001000	128
010111	192	000011001001	192
0110111	256	000001011011	256
00110110	320	000000110011	320
00110111	384	000000110011	384
01100100	448	000000110101	448
01100101	512	0000001101100	512
01101000	576	0000001101101	576
01100111	640	0000001001010	640
011001100	704	0000001001011	704
011001101	768	0000001001100	768
011010010	832	0000001001101	832
011010011	896	0000001110010	896
011010101	960	0000001110011	960
011010101	1024	0000001110100	1024
011010110	1088	0000001110101	1088
011010111	1152	0000001110110	1152
011011000	1216	0000001110111	1216
011011001	1280	0000001010010	1280
011011010	1344	0000001010011	1344
011011011	1408	0000001010100	1408
010011000	1472	0000001010101	1472
010011001	1536	0000001011010	1536
010011010	1600	0000001011011	1600
011000	1664	0000001100100	1664
010011011	1728	0000001100101	1728
000000000001	EOL	000000000001	EOL

Fig.V.2 - Formação da Tabela de Composição.

Fig.V.3 - Formação da Tabela de Terminação

PALAVRA CÓDIGO	CORRENTE BRANCA	PALAVRA CÓDIGO	CORRENTE PRETA
00110101	0	0000110111	0
000111	1	010	1
0111	2	11	2
1000	3	10	3
1011	4	011	4
1100	5	0011	5
1110	6	0010	6
1111	7	00011	7
10011	8	000101	8
10100	9	000100	9
00111	10	0000100	10
01000	11	0000101	11
001000	12	0000111	12
000011	13	00000100	13
110100	14	00000111	14
110101	15	0000011000	15
101010	16	00000010111	16
101011	17	00000011000	17
0100111	18	00000001000	18
0001100	19	000001100111	19
0001000	20	000001101000	20
0010111	21	000001101100	21
0000011	22	000000110111	22
0000100	23	000000101000	23
0101000	24	000000010111	24
0101011	25	000000011000	25
0010011	26	0000011001010	26
0100100	27	0000011001011	27
0011000	28	0000011001100	28
00000010	29	0000011001101	29
00000011	30	0000001101000	30
00011010	31	0000001101001	31
00011011	32	0000001101010	32
00010010	33	0000001101011	33
00010011	34	0000011010010	34
00010100	35	0000011010011	35
00010101	36	0000011010100	36
00010110	37	0000011010101	37
00010111	38	0000011010110	38
00101000	39	0000011010111	39
00101001	40	0000001101100	40
00101010	41	0000001101101	41
00101011	42	0000011011010	42
00101100	43	0000011011011	43
00101101	44	0000001010100	44
00000100	45	0000001010101	45
00000101	46	0000001010110	46
00001010	47	0000001010111	47
00001011	48	0000001100100	48
01010010	49	0000001100101	49
01010011	50	0000001010010	50
01010100	51	0000001010011	51
01010101	52	0000000100100	52
00100100	53	0000000110111	53
00100101	54	0000000111000	54
01011000	55	0000000100111	55
01011001	56	0000000101000	56
01011010	57	0000001011000	57
01011011	58	0000001011001	58
01001010	59	0000000101011	59
01001011	60	0000000101100	60
00110010	61	0000001011010	61
00110011	62	0000001100110	62
00110100	63	0000001100111	63

V.2 - Rotina de Inicialização

```
STRT:   XOR AX, AX   ; zerar AX
        XOR BX, BX
        XOR CX, CX
        XOR DX, DX
        XOR DI, DI
        XOR BP, BP
        XOR SI, SI   ; zerar SI
        jmp inic
INIC:   MOV DX, 4000H ;Inicializar todos os segementos
        MOV ES, DX    ; de trabalho, ponteiros e as valores
        MOV DX, 3000H ; iniciais de cada posição de memória.
        MOV DS, DX
        NOP
        MOV DX, 3000H
        MOV DS, DX
        NOP
        MOV [1500], AX
        MOV [1508], AX
        MOV [1520], AX
        MOV [1540], AX
        MOV AX, 6800H
        MOV SI, AX
        MOV [1510], AX
        MOV WORD PTR [SI], 0000H
        MOV AX, 6000H
        MOV [1526], AX
        MOV AX, 000FH
        MOV [1530], AX
        MOV AX, 0004H
        MOV [1534], AX
        MOV AX, 4000H
        MOV [1522], AX
        MOV AX, 2000H
        MOV BP, AX
        MOV BX, 2000H
        MOV SI, 6800H
        MOV WORD PTR [SI], 0000H
```

```
MOV SI, 0000H
MOV AX, ES: [SI] ; Traz a palavra código p/ AX
PUSH AX
AND AX, 0FFFH
CMP AX, 0800H ; Procura FDL
JNZ ERRO  ;Não achou FDL
MOV CL, 0CH
POP AX
SHR AX, CL
MOV [1538] , AX
```

A função desta rotina é inicializar o programa principal de decodificação. A tarefa inicial é zerar os registradores básicos do microcomputador como AX, BX, CX, DX, DI, BP e SI, através da instrução XOR, que é mais rápida em comparação às outras instruções. Devido à importância do tempo de execução nas aplicações em tempo real, é imprescindível a escolha pelas instruções de tempo mínimo, tal que o programa satisfaça as especificações de desempenho requeridas na aplicação.

Depois de zerar os registradores, a rotina define um registrador para indicar o segmento de trabalho. Neste caso, a rotina escolhe o registrador ES para indicar a localização dos segmentos de dados e dedica o segmento 4000:0000h para ser o receptor de dados da imagem. Como o tamanho do arquivo de dados pode ultrapassar vários segmentos, escolhe-se o segmento 3000:0000h para ser a área que irá armazenar as tabelas de códigos pretos e brancos e os ponteiros do pré-buffer de impressão. Note-se que o registrador DS irá sempre indicar o segmento onde se guarda as tabelas de códigos pretos e brancos. A figura V.4 resume a organização deste programa no que se refere à colocação dos dados iniciais em todos os ponteiros e respectivas posições na memória.

Após o processo de inicialização, inicia-se o processo de decodificação que é descrito a seguir.

Fig.V.4 - Organização da Memória do Programa de Decodificação

SEGMENTO OFFSET	EXPLICAÇÃO "CONTEÚDO	TAMANHO	ESTADO INICIAL
3 000 : 1 500	Contador de 8 linhas	byte	zero
3 000 : 1 508	Contador do total de linhas	word	zero
3 000 : 1 510	Ponteiro de início do pré-buffer de impressão		
3 000 : 1 520	Ponteiro de word atual no buffer de recepção	word	zero
3 000 : 1 522	Ponteiro do segmento atual	word	
3 000 : 1 528	Ponteiro do buffer de impressão		
3 000 : 1 530	Ponteiro do bit atual dentro de um word no buffer de recepção	word	15 "F"
3 000 : 1 534	Contador de bits já separados para processamento	byte	4
3 000 : 1 538	Local oara guardar resultado da máscara	word	
3 000 : 1 540	Contador do total de bits por linha ou contador de 2048 bits	word	zero
3 000 : 2 000	Tabela de códigos bracos		
3 000 : 4 000	Tabela de códigos pretos		
3 000 : 6 000	Início do buffer de impressão		
3 000 : 6 800	Início do pré-buffer de impressão		
4 000 : 0 000	Buffer de recepção de dados recebidos		

V.3 - ROTINA CONSU

```
CONSU: MOV AX, BP          ; Consulta a tabela de códigos
       ADD AX, WORD PTR [1538]
       MOV SI, AX
       MOV AX, [SI]
       SHR AL, 1
       JZ ZERO
       JNC TERM            ; Dados são de terminação
       AND AX, 001FH ; Processar os dados de composição
       MOV CL, 06H
       SHL AX, CL
       MOV WORD PTR DS : [1510], AX
       MOV AX, BP
       ADD SI, AX
       MOV AX, [SI]
       JMP  COMEÇO
```

Após a deteção do código FDL, desloca-se o conteúdo de AX doze vezes para a direita para obter os quatro bits restantes da primeira leitura do buffer de recepção. Neste caso, considera-se os quatro bits porque o primeiro código após FDL é o código de pontos brancos tendo um número mínimo de quatro bits. Em seguida, salva-se o conteúdo do registrador AX para poder processar os próximos códigos. Se, da leitura de um código de quatro bits, não for encontrado o número de pontos brancos correspondente na tabela de consulta, é preciso salvar os quatro bits e realizar a leitura de mais um bit. Salva-se de novo o conteúdo do registrador AX e repete-se o processo até encontrar o número de pontos brancos correspondente ao código.

Neste procedimento, conforme já mencionado anteriormente, o próprio código serve como endereço para a tabela de consulta. Quando forem encontrados os dados válidos no endereço especificado pelo registrador SI, eles são transferidos para uma área de memória denominada Pré-buffer de impressão. Esses dados podem ser de dois tipos, ou seja, ou são dados de composição (64 pontos ou mais) ou são dados de terminação (menos de 64 pontos). No caso de composição, mascara-se o conteúdo do registrador AX com uma máscara adequada de tamanho variável para separar os dados corretos de composição, multiplica-

se o valor encontrado por 64 e leva-se o resultado para o pré-buffer de impressão.

V.4 - ROTINA TERM

```
TERM : AND AX, 003FH ; Processar os dados de Terminação
       MOV SI, [1510]
       ADD AX, [SI]
       MOV WORD PTR DS : [SI], AX
       MOV BX, [1540]
       ADD AX, BX
       MOV [1540], AX
       CMP AX, 0800H
       JE EXFDL; Desvia para rotina EXFDL
       INC SI
       INC SI
       MOV WORD PTR [SI], 0000H
       MOV AX, BP
       XOR AX, 6000H
       MOV BP, AX
       JMP COMEÇO
```

No caso do código de terminação, usando uma máscara apropriada para separar os dados corretos de terminação, soma-se o número de pontos de terminação com o número de pontos de composição e o resultado é levado para o pré-buffer de impressão. Em seguida, procura- se o código FDL, muda a consulta para a tabela de pontos pretos e continua o processo de decodificação.

V.5 - ROTINA COMEÇO

```
COMEÇO: XOR AX, AX
        MOV [1534], AX ; Zera o ccontador bits usados
        MOV [1538], AX
        MOV AX, BP
        CMP AX, 2000H
        JE BRANCO
        CALL PEGMB ; Pega mais um bit para ser
        CALL PEGMB ; Processada e consulta tabela
```

```
        JMP CONSU
BRANCO: CALL PEGMB ; Começa uma linha nova, então
        CALL PEGMB ; pega mais 4 bits
        CALL PEGMB
        CALL PEGMB
        JMP CONSU
```

A sua função básica é atualizar alguns ponteiros, verificar a sua posição correta no processo de decodificação para confirmar se o processamento está sendo feito nas correntes de pontos brancos ou de pontos pretos. Caso seja nas correntes de pontos brancos, desvia-se o controle para a rotina BRANCO para pegar mais quatro bits buffer de recepção e consultar a tabela de pontos brancos. Caso contrário, pega-se mais dois bits do buffer de reccepção e consulta a tabela de pontos pretos.

V.6 - Rotina MUDAR

```
MUDAR: MOV AX, 2000H ; Começa uma linha nova
       MOV BP, AX
       MOV WORD PTR [1540], 0000H
       JMP CONSU
  TOT: INC WORD PTR [1508] ; totalizador de linhas
       INC WORD PTR [1500]
 OITO: MOV AX, DS: [1500] ; Chegou oito linhas
       CMP AX, 08H        ; então monta buffer de impressão
       JNZ MUDAR          ; e imprime os oito linhas
       CALL MONBUF
       XOR AX, AX
       MOV [1500], AX
       JMP MUDAR
```

A função desta rotina é alterar a consulta da tabela de códigos pretos para a de códigos brancos ou vice-versa. No começo de cada linha, é necessário uma consulta à tabela de códigos brancos para para cumprir as exigências da norma CCITT

V.7 - ROTINA ZERAB

```
ZERAB: MOV DI, 6000H ; Zerar pré-buffer de impressão
       XOR AX, AX
 LOOP: MOV [DI], AX
       INC DI
       CMP DI, 6800H
       JNZ LOOP
       RET
```

Esta rotina zera ou limpa a área de memória denominada buffer de impressão. Ela vai zerando, byte por byte, desde o endereço inicial do buffer de impressão até o seu endereço final

V.8 - ROTINA ZERO/FDL

```
ZERO:  MOV AX,[1534] ; Processar os dados no caso de que
       CMP AX, 0DH ; os dados não são validos
       JE ERRO
       CALL PEGMB
       MOV AX, [1534]
       CMP AX, 0CH
       JE FDL
       JMP CONSU
 FDL:  MOV AX, DS :[1538] ; Processar os dados no caso de
       AND AX, 0FFFH ; que os dados não são validos
       CMP AX, 001H
       JE TOT
       JMP CONSU
       INC WORD PTR [1540]
       CMP WORD PTR [1540], 0810
       JE MUDAR
       JMP CONSU
       RET
```

Quando se utiliza a rotina CONSU para utilizar a tabela de consulta de códigos e encontra dados inválidos, o controle do processo passa para a rotina ZERO. Nesta rotina, verifica-se o número de bits que já foram usados. Lembrando-se que o número máximo de bits de qualquer código é 13 , quando o número verificado é 13 e não

encontra dados válidos na posição de memória endereçada pelo próprio código, esta rotina transfere o controle para rotina para ERRO para gerar uma mensagem de erro. Por outro lado, quando o número verificado não for 13 , então pega mais um bit e analisa os dados. Se encontrar o código FDL, o controle é passado para rotina FDL; caso contrário, consulta-se a tabela de código através da rotina CONSU.

A função básica deste rotina é procurar o código FDL. Para isso, esta rotina pega os primeiros 12 bits da registrador AX, e os compara com o código FDL. Se os dados forem do código FDL, ela transfere o controle para a rotina totalizadora de linhas, senão transfere o controle para a tabela de consulta.

V.9 - ROTINA PEGMB

```
PEGMB: INC BYTE PTR [1530] ; Rotina que pega mais um bit
       CMP BYTE PTR [1530], 10H ; da buffer da recepção
       JB MB              ; para ser processada
       MOV AL, 00H
       MOV [1530], AL
       INC WORD PTR [1520]
       INC WORD PTR [1520]
       JNZ MB
       MOV AX, ES
       ADD AX, 1000H
       MOV ES, AX
   MB: INC BYTE PTR [1534]  ; O bit é zero "0"
       MOV SI, [1520]
       MOV AX, 0001H
       MOV CL, [1530]
       SHL AX, CL
       AND AX, ES : [SI]
       JNZ OK
       RET
   OK: MOV CL, [1534] ; O bit é um "1"
       DEC CL
       MOV AX, 0001H
       SHL AX, CL
       OR [1538], AX
```

```
        RET
```

Esta rotina tem por função ler mais um bit para ser processado. Ela inicia apontando para o bit atual do buffer de recepção para que este seja processado dentro de uma palavra "word" escolhida. Depois, verifica-se foi alcançado o último bit da palavra atual através de uma comparação com o conteúdo de um ponteiro com 16, em decimal.

A idéia básica desta rotina é apontar para um "word" no buffer de recepção e, dentro deste "word", apontar para o bit atual, o qual deve ser levado para o processamento. Quando termina o processamento dos 16 bits de uma palavra, aponta-se para a próxima palavra. O resultado do teste de conteúdo do ponteiro (1530^{H}) é analisado para verificar se todos os 16 bits da palavra já foram testadas. Em caso afirmativo, zera-se o ponteiro do bit atual (1530^{H}) e incrementa-se o conteúdo do ponteiro do "word" atual para apontar para o próximo "word" no buffer de recepção. Além disso, faz-se um outro teste para verificar se o segmento acabou. Normalmente, quando o segmento chega ao fim, o endereço é zerado, ou seja, se o ponteiro do "word" atual (1520^{H}) mostrar o valor 0000, significa que o segmento acabou. Neste caso, coloca-se o valor do segmento no acumulador e soma-se um 1000h para apontar para o próximo segmento.

No final, realiza-se um teste para verificar se os 16 bits do "word" atual foram processados. Caso negativo, incrementa-se o ponteiro que guarda a quantidade de bits que já foram usadas e armazena o conteúdo do ponteiro do "word" atual no registrador SI para dar continuidade ao processamento normal.

V.10 - ROTINA EXFDL

```
EXFDL: XOR AX, AX  ; Rotina que extrai o sinal de
       MOV [1534], AX ; Fim da Linha (FDL)
       MOV BX, 000CH
       MOV [1538], AX
  PEG: CALL    PEGMB
       DEC BX
       JNZ PEG
```

```
        MOV AX, [1538]
        CMP AX, 3800H
        JZ  TOT
        AND AX, AX
        JNZ TOT
PROC1:  XOR BX, BX
  VLT:  XOR AX, AX
        MOV [1534], AX
        DEC BX
        JZ TOT
        CALL   PEGMB
        AND AX, AX
        JZ  VLT
        JMP TOT
ERRO1:  MOV AX, [1538]
        CMP AX, 1000H
        JZ  TOT
        CALL PEGMB
        AND AX, AX
        JZ PROC1
        JMP ERRO ; Escreve transmissão ruim
 ZERO:  MOV AX, [1534]
        CMP AX, 0DH
        JE      ERRO1
        CALL   PEGMB
        MOV AX, [1534]
        CMP AX, 0CH
        JE FDL
        JMP CONSU
```

Esta rotina deve procurar e extrair o código FDL, ou seja, considera-se 12 bits para processamento e analisa-os para verificar se existe onze "0" e um "1" (0000 0000 0001). O código FDL não é imprimido pois êle é apenas um sinal de sincronismo para definir o começo de uma página de informação gráfica e o fim de uma linha de informação gráfica.

V.11 - ROTINA ERRO

```
 ERRO: MOV AX, [1538]
       CMP AX, 1000H
       JZ TOT
       AND AX, AX
       JZ   PROC1
       MOV AX, 0F
       MOV BP, AX
 LIMP: MOV CX, 0000H; Limpar a tela
       MOV DX, 1E4FH
       MOV BH, 07H
       MOV AX, 0600H
       INT 21
CENTR: MOV DX, 000CH; Centalizar o cursor
       MOV BH, 001DH
       MOV AH, 02H
       INT 21
 DISP: MOV AH, 09H ; Mostra mensagen de erro
       MOV DL, 54H ; "T"
       INT 21
       MOV AH, 09H
       MOV DL, 52H ; "R"
       INT 21
       MOV AH, 09H
       MOV DL, 41H ; "A"
       INT 21
       MOV AH, 09H
       MOV DL, 4EH ; "N"
       INT 21
       MOV AH, 09H
       MOV DL, 53H ; "S"
       INT 21
       MOV AH, 09H
       MOV DL, 4DH ; "M"
       INT 21
       MOV AH, 09H
       MOV DL, 49H ; "I"
```

```
        INT 21
        MOV AH, 09H
        MOV DL, 53H ; "S"
        INT 21
        MOV AH, 09H
        MOV DL, 53H ; "S"
        INT 21
        MOV AH, 09H
        MOV DL, 41H ; "A"
        INT 21
        MOV AH, 09H
        MOV DL, 4FH ; "O"
        INT 21
SPACE:  MOV AH, 09H ; Enviar tres espaços entre
        MOV DL, 20H ; as duas palavras de mensagem
        INT 21
        MOV AH, 09H
        MOV DL, 20H
        INT 21
        MOV AH, 09H
        MOV DL, 20H
        INT 21
        MOV AH, 09H
        MOV DL, 52H ; "R"
        INT 21
        MOV AH, 09H
        MOV DL, 55H ; "U"
        INT 21
        MOV AH, 09H
        MOV DL, 49H ; "I"
        INT 21
        MOV AH, 09H
        MOV DL, 4DH ; "M"
        INT 21
        DEC BP
        JNZ   LIMP
        RET
```

Em tôdas as páginas ou em todos os documentos codificados, o primeiro código que aparece é o código FDL. Assim, quando o programa de decodificação analisa os primeiros 12 bits e encontra o código FDL, o processo de decodificação ocorre normalmente. Mas, quando isto não acontece, é julgado que houve um erro de transmissão e é enviada uma mensagem para avisar o operador do aparelho FAX. Neste caso, a rotina ERRO limpa a tela, centraliza o cursor na tela para que a mensagem de erro seja apresentada.

V.12 - ROTINA MONBUF

```
MONBUF: CALL   ZERAB ; Zera o buffer de impressão
        MOV BL, 80H
        MOV SI, 6800H
  PAG:  MOV DI, 6000H ; Monta o pré-buffer de impressão numa
 VOLTB: MOV AX, [SI]  ; forma adequada para impressão
        INC SI
        INC SI
        ADD DI, AX
        CMP DI, 67FF
        JA   NOVAG
        MOV CX, [SI]
        INC SI
        INC SI
 LOOP:  OR [DI], BL ; Prepara mascara correta  para
        INC DI      ; pegar o bit esolhida
        INC DI
        DEC CX
        JNZ LOOP
        CMP DI, 67FF
        JA   NOVAG
        JMP VOLTB
NOVAG:  SHR BL, 1 ; Pega próxima linha do pré-buffer de
        JNC PAG   ; impressão
        CALL  PRNT; Imprimir as linhas prontas
        RET
```

Os dados que estão armazenados no pré-buffer de impressão não estão prontos para a impressão, necessitando para tal realizar algumas modificações. Isto porque os dados no pré-buffer de impressão representam o número de pontos brancos e pretos, requerendo uma transformação em UM'S ou ZERO'S.

Após a transformação do número de pontos em valores binários, é preciso agrupá-los a cada oito linhas de modo que o primeiro bit das oito linhas resulte no primeiro byte, o segundo bit das oito linhas em segundo byte e assim por diante, para transformar as oito linhas em 2048 bytes ou 2048 colunas gráficas. Finalmente, cada byte precisa ser convertida para o seu equivalente em sistema de numeração decimal antes de enviá-los para a impressão.

V.13 - ROTINA PRNT

```
PRNT: MOV AH, 05H ; Cancelar qualquer programação
      MOV DL, 1BH ; anterior da impressora
      INT 21
      MOV AH, 05H
      MOV DL, 40H
      INT 21
      MOV AH, 05H ; Entrar no modo gráfico
      MOV DL, 1B
      INT 21
      MOV AH, 05
      MOV DL, 4B
      INT 21
      MOV AH, 05H ; Definição de quantos bytes
      MOV DL, 00H ; serão enviados para a impressora
      INT 21
      MOV AH, 05H
      MOV DL, 08H
      INT 21
      MOV SI, 6000H
      MOV BX, 6800H
  PB: MOV AH, 05H  ; Coloca os dados de pré-buffer
      MOV DL, [SI] ; de impressão em buffer
```

```
INT 21        ; de impressão
INC SI
CMP SI, BX
JNZ   PB
MOV AH, 05H ; Ativar o modo de microespaçamento
MOV DL, 1BH ; para colocar as próximas 8 linhas
INT 21
MOV AH, 05H
MOV DL, 33H
INT 21
MOV AH, 05H
MOV DL, 08H
INT 21
RET
```

Esta rotina é usada para imprimir os dados do arquivo de imagem. Ela inicializa a impressora, limpa o seu buffer de recepção e cancela qualquer programação feita anteriormente. Isso é feito através de uma instrução que habilita uma porta de comunicação com a impressora, ou seja, sempre que for estabelecer uma comunicação com a impressora ou imprimir alguns caracteres, deve-se armazenar um valor "05H" no registrador AH, seguido pelo caracter ou comando que se deseja imprimir e chama-se a interrupção "INT 21".

O próximo passo é entrar no modo gráfico o qual possibilita a impressão das imagens de FAX através da instrução ESC K. Estes dois caracteres são enviados um após o outro, ou seja, primeiramente o código de controle ESC e posteriormente o caracter K.

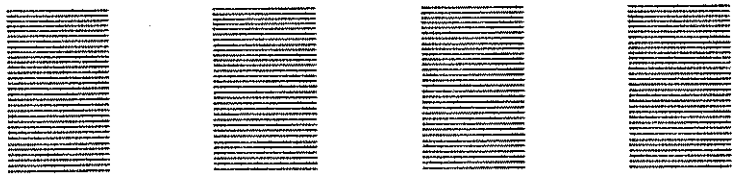
Existem várias densidades de impressão disponíveis pela impressora. Entre elas, foram testadas as densidades L, Y, K e Z usando alguns padrões em xadrez de tamanhos 32x32, 64x64, 128x128, 256x256 e 512x512 para selecionar a densidade mais adequada para a impressão de arquivos FAX neste trabalho.

A densidade K é de 60 pontos/polegada (ppp) na horizontal e de 72 ppp na vertical. Compondo uma página inteira de 297 mm x 210 mm (tamanho A_4), a resolução máxima que se consegue é de 504 pontos na horizontal e de 855 pontos na vertical, com 351 pontos a mais na direção vertical que distorcem a imagem de FAX reproduzida. Essa distorção pode ser eliminada ajustando a distância entre as linhas verticais e horizontais através das instruções de micro-

espaçamento. Esta densidade é acionada através da instrução 1B 2A 00 ou 1B 4B nH1 nH2, onde 1B é ESC, 2A é "*", 00 é NUL e 4B é "K".

Os testes realizados com as outras densidades (L,Y e Z) mostraram que elas são inadequadas para imprimir as imagens de FAX porque os pontos ficam muito próximos um do outro

Outras densidades mais altas possuem problemas de definição de pontos,pois o tamanho das agulhas é muito grande. Portanto, este tamanho não permite imprimir mais pontos na mesma área, pois os pontos ficam excessivamente próximas uns dos outros e a imagem fica irreconhecível. Os resultados primários obtidos são ilustrados na figura V.5.



[K]

[L]

[Y]

[Z]

Fig.V.5 - Ilustração da impressão para densidades K,L,Y,Z

VI.1 - ANÁLISE DOS CÓDIGOS DE HUFFMAN MODIFICADO

Neste trabalho foram estudadas várias combinações de formatos de imagem de fac-símile e montadas várias densidades de uma linha de informação e uma página inteira. Para uma página inteira de informação sem codificação, é necessária uma área de memória de 250 K bytes, enquanto que para uma página codificada o tamanho da memória alocada varia de 3 Kbytes a 1 Mbytes.

Os resultados da compressão que variaram de -300% a 98,8% são apresentados na Tabela VI.1. Ela evidencia que, se a variação da informação for muito intensa, o código não é eficiente. Este é o caso de uma imagem formada por uma sequência de um ponto preto e um ponto branco, cuja compressão é de aproximadamente -300% , ou seja, é requerida uma área de memória quatro vezes maior do que aquela necessária para transmitir uma mesma página sem qualquer codificação.

Tabela VI.1 - Resumo da Taxa de Compressão para vários padrões

PADRÃO	% COMPRESSÃO	FATOR DE COMPRESSÃO
1B 1P	-300%	-4 vezes
8B 8P	≅ 31%	≅ 1,4 vezes
16B 16P	≅ 50%	≅ 2 vezes
32B 32P	≅ 68%	≅ 3,1 vezes
64B 64P	≅ 87%	≅ 7,3 vezes
126B 126P	≅ 91%	≅ 11,4 vezes
256B 256P	≅ 94%	≅ 16,6 vezes
512B 512P	≅ 95,6%	≅ 22,7 vezes
1024B 1024P	≅ 98%	≅ 50 vezes
1728P(página preta)	≅ 98,4%	≅ 62,5 vezes
1728B(página branca)	≅ 98,8%	≅ 83,3 vezes

À medida que a semelhança entre um ponto da imagem e um ponto vizinho aumenta, a eficiência do código também aumenta. Em outras palavras, à medida que o tamanho das correntes de pontos pretos ou brancos aumenta, a compressão é maior. Por exemplo, a compressão de uma imagem formada por uma corrente de 8 pontos brancos seguida por uma corrente de 8 pontos pretos é menor do que a de uma imagem formada por correntes de 16 pontos brancos seguidos por 16 pontos pretos. No primeiro caso, a compressão é cerca de 31% e, no segundo caso, é cerca de 50% .

Para uma ilustração dessa tabela , considera-se o caso de uma folha composta de uma sequência de 16 pontos brancos e 16 pontos pretos codificados.

Código de 16 pontos brancos = 101010 / Comprimento = 6 bits
Código de 16 pontos pretos = 0000010111 / Comprimento = 10 bits

Uma linha original contém 1728 pontos, donde se tem 54 correntes de 16 pontos brancos e outras 54 correntes de 16 pontos pretos. Assim, uma linha codificada mais os 12 bits/linha para sincronismo, requer:

Nº de bits/linha codificada = 54x6 + 54x10 + 12 = 876 bits

Assim, para uma página inteira com 1147 linhas codificadas juntamente com os 12 bits FDL da 1ª linha e mais 5 FDL da última linha, tem-se:

Nº de bits/página codificada = 12 + 876x1147 + 5x12 = 1004844 bits
≅ 125605 bytes ≅ 125 Kbytes

Portanto, dado que uma página sem codificação requer 250 Kbytes, a porcentagem de compressão pela codificação será :

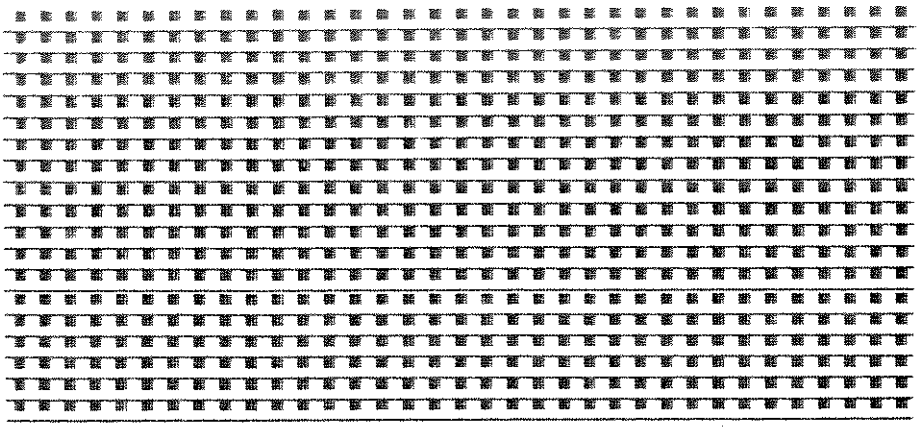
$$\% \text{ compressão} \cong \left(1 - \frac{125K}{250K} \right) \times 100\% = 50\%$$

VI.2 - Análise dos Resultados da Impressão

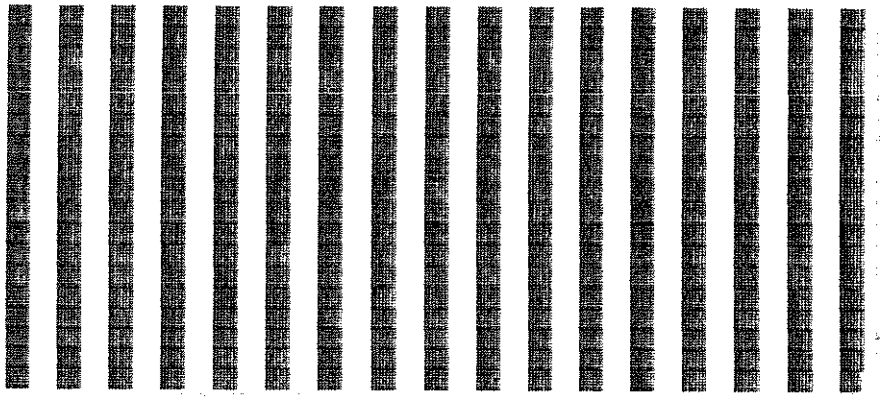
Pode-se observar que existem algumas falhas em uma ou mais linhas na direção vertical. Na direção horizontal, pode-se observar que a impressão é consistente e sem falhas de impressão. Isto mostra que o motor de passo que controla o carro de impressão na direção horizontal está em condições normais de funcionamento, enquanto que o motor de passo que controla a rotação do cilindro na direção vertical está com algum defeito que vai ser analisado, no futuro próximo. Observando as figuras abaixo [A] a [H] podem notar as falhas na direção vertical .estas figuras mostra a impressão vários quadrados, o que revela claramente as falhas da impressora. O problema de falhas não foi estudado ainda, mas é provavél que esteja relacionado ao hardware dos três sistemas mecânicos da impressora.

As figuras [A] a [H] são amostras de resultados obtidos,e são compostos de seguintes forma:

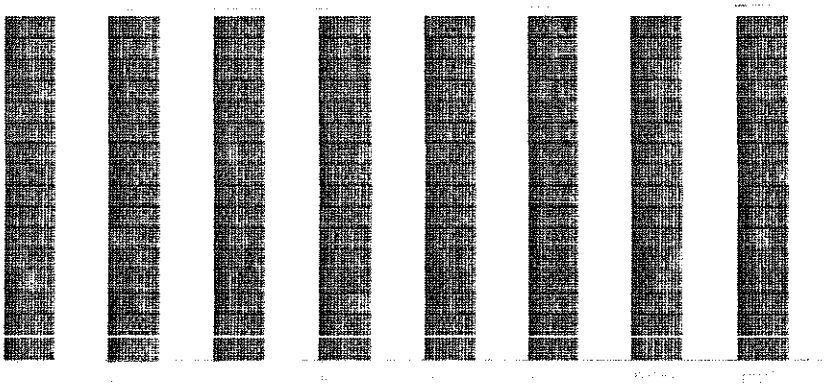
[A] Formado por 4 pontos Brancos seguido por 4 pontos Pretos.
[B] " " 8 " " " " 8 " "
[C] " " 16 " " " " 16 " "
[D] " " 32 " " " " 32 " "
[E] " " 64 " " " " 64 " "
[F] " " 128 " " " " 128 " "
[G] " " 256 " " " " 256 " "
[H] " " 512 " Pretos



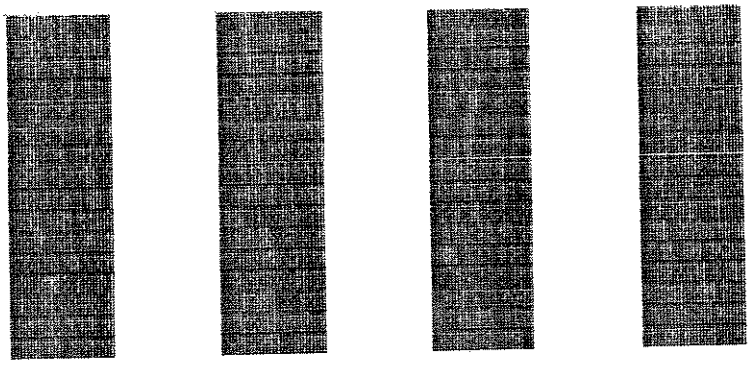
[A]



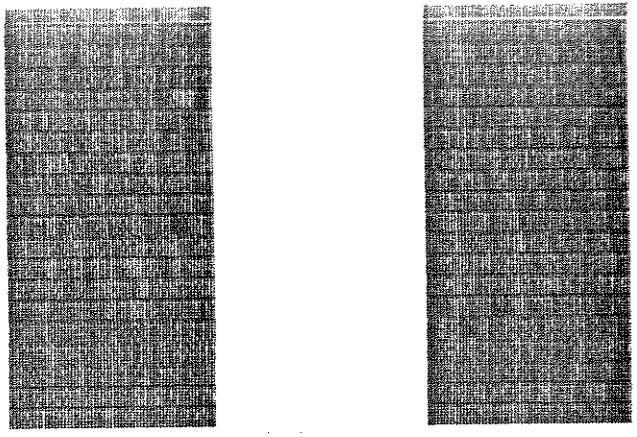
[B]



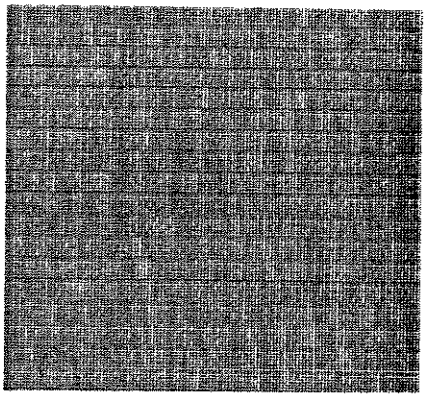
[C]



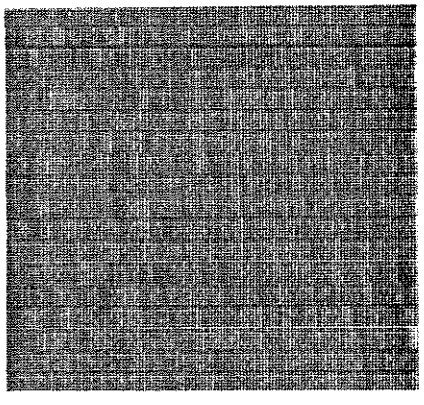
[D]



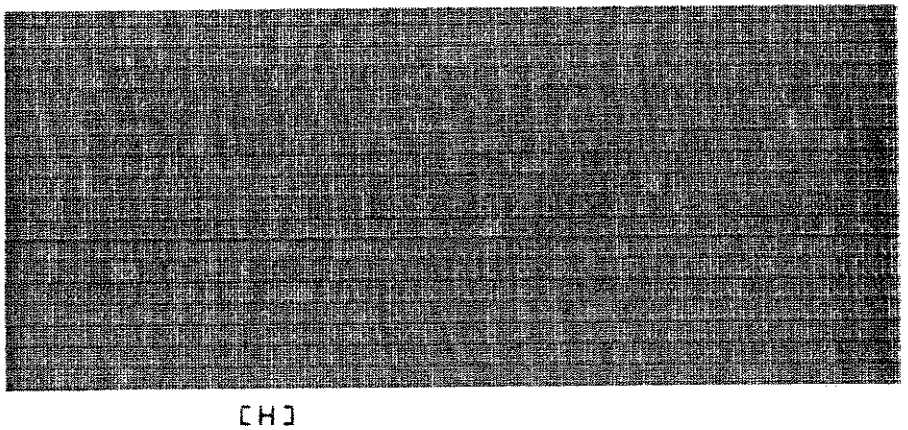
[E]



[F]



[G]



[H]

VI.3 - Conclusão

A análise dos resultados mostrou que é viável obter uma reprodução de boa qualidade utilizando o esquema proposto neste projeto. Os tempos de codificação e de decodificação estão dentro das especificações exigidas pelos padrões internacionais do CCITT.

Quanto à resolução, os resultados obtidos mostram que a ela é de boa qualidade e é possível melhorá-la utilizando impressoras de melhor qualidade como, por exemplo, impressora de nove agulhas ou de 24 agulhas, impressora laser ou jato de tinta. A falha de impressão observada no sentido vertical não é um problema no sentido de piorar a qualidade de impressão, até o ponto de ter problemas com o reconhecimento de detalhes da imagem transmitida.

De um modo geral, as imagens de teste recebidas e imprimidas são de boa qualidade e confiáveis, para as aplicações comerciais. O esquema proposto produz uma qualidade aceitável de imagem e, portanto, é possível utilizá-lo como uma alternativa para o aparelho FAX de baixo custo, em comparação com os aparelhos FAX comerciais existentes no mercado. Outra vantagem deste esquema proposto é a facilidade de manutenção, visto que o esquema proposto não contém circuitos integrados complexos ou sofisticados, nem circuitos integrados dedicados, o que torna a manutenção complicada e altamente especializada.

Outro fator relevante deste projeto é a possibilidade de oferecer ao usuário de uma máquina FAX, uma alternativa simples e barata. Porém, esta simplicidade de projeto traz algumas limitações tais como a falta de flexibilidade quanto à qualidade de impressão. A impressora matricial tem densidades fixas de impressão e isto limita o número de pontos por linha de impressão. As densidades possíveis de impressão aceitável que ela oferece são de 60, 72, 80, 90, 120, 144 e 240 ppp. Caso seja necessário uma melhor definição, então a solução é desenvolver um sistema de FAX utilizando uma impressora de alto desempenho, como impressoras laser ou jato de tinta que ofereccem uma resolução de 300 ppp ou mais.

O esquema de codificação segue a padrão internacional sugerido pelo CCITT. Ele foi testado por várias agências internacionais de telecomunicações e foi escolhido para ser padronizado para uso em todas as máquinas FAX do grupo G_3/CCITT. Então, o código é eficiente e apresenta uma compressão muito grande dos arquivos de transmissão originais, podendo diminuir o tempo de transmissão e, consequentemente economizando o custo de envio de documentos gráficos pela linha telefônica.

VI.4 - Propostas para Trabalhos Futuros

As propostas apresentadas abaixo são no sentido de melhorar a qualidade da imagem de FAX imprimida, aumentar a eficiência do sistema de codificação e testar a transmissão de imagem FAX à distância.

01) Modificar os programas de codificação e decodificação para aproveitar a correlação entre dados de FAX na linha horizontal e na linha vertical (correlação bidimensional).

02) Modificar os programas de codificação e decodificação de imagens coloridas.

03) No projeto atual, utilizou-se uma impressora matricial. Pode-se modificar a rotina de impressão para a utilização de impressoras de alta resolução como a impressora de 24 agulhas, a impressora laser ou jato de tinta.

04) No projeto atual, fez-se uma decodificação local das imagens de FAX. Implementar nos projetos futuros, alguns protocolos de transmissão e recepção de imagens de FAX à distância.

Bibliografia

[1] Eng. Luis Benedito Cypriano , Microprocessador Z80 Software, Vol. 2, 3a Edição, Livros érica Editora Ltda. 1986

[2] Fax- CCITT Blue Book Vol., VII- Fascicle , V11.3 Rec. T.0-T.63 Nov/1988.

[3] RT -73 Topicos Sobre Fac-Simile , Contrato Telebrás 139/76, Relatorio ténico , Junho/1980.

[4] N.S. Jayant- Peter Noll, Digital Coding of Waveforms, Prentice-Hall, Signal Processing Series 1984.

[5] Yasuda et al, Advances in Fax, Proceedings of the IEEE , Vol. 73 No.4 April 1985.

[6] Robert E. Krallinger , Evolution of Standard for Business Facsimile, IEC, 1978.

[7] Rima Impressoras S.A. , Manual de Instalação e Operação: Impressora matricial RIMA XT 180, Rima Impressoras S.A., Dez/1988.

[8] Scopus Tecnologia S.A., NEXUS- 2600 Referência do BASIC, Scopus tecnologia S.A., Agosto/1986.

[9] HUFFMAN, D.A., A method for the construction of Minimum Redundancy Codes .Proc. IRE, vol. 40, No. 10 set. 1952.

[10] KELLEY, J.E. JR; IBM PC E SEUS COMPATíVEIS ; Trad.e rev. de Rezende, J.D.e Perez, C.M., McGraw-Hill Ltd, 1987.

[11] MORGAN, C.L.& WAITE, M.; 8086/8088-Manual do Microprocessador de 16 bits. Trad. e rev. de Ungnius, L.G.E. e Silva Neto, U.R., McGraw-Hill, 1988.

[12] ROLLINS, D.; IBM-PC 8088 Macro Assembler Programming.; Macmillan Publishing Company, 1985.

[13] ABEL, P.; Assembler for the IBM-PC and PC-XT; Reston Publishing Company, 1984.

[14] RODNAY ZAKS; Microprocessor from chip to systems ; SYBEX Inc ,1977

[15] ENG. ANTONIO VISCONTI; Microprocessadores 8080 e 8085 SOFTWARE; Vol. 2 ; LIVROS éRICA EDITORA LTDA. 1984

APêNDICE

Apéndice A - Listagem completa do Programa de Codificação

```
-
-u100 160
15B6:0100 31C0            XOR     AX,AX
15B6:0102 31DB            XOR     BX,BX
15B6:0104 31C9            XOR     CX,CX
15B6:0106 31D2            XOR     DX,DX
15B6:0108 31FF            XOR     DI,DI
15B6:010A 31ED            XOR     BP,BP
15B6:010C 31F6            XOR     SI,SI
15B6:010E 1E              PUSH    DS
15B6:010F 90              NOP
15B6:0110 B80030          MOV     AX,3000
15B6:0113 8ED8            MOV     DS,AX
15B6:0115 90              NOP
15B6:0116 31C0            XOR     AX,AX
15B6:0118 BFFEFF          MOV     DI,FFFE
15B6:011B 4F              DEC     DI
15B6:011C 4F              DEC     DI
15B6:011D 8905            MOV     [DI],AX
15B6:011F 83EF04          SUB     DI,+04
15B6:0122 8905            MOV     [DI],AX
15B6:0124 47              INC     DI
15B6:0125 47              INC     DI
15B6:0126 B80040          MOV     AX,4000
15B6:0129 8905            MOV     [DI],AX
15B6:012B BB1000          MOV     BX,0010
15B6:012E BF0080          MOV     DI,8000
15B6:0131 31F6            XOR     SI,SI
15B6:0133 E8FA01          CALL    0330
15B6:0136 90              NOP
15B6:0137 90              NOP
15B6:0138 8B04            MOV     AX,[SI]
15B6:013A 46              INC     SI
15B6:013B 46              INC     SI
15B6:013C D1E8            SHR     AX,1
15B6:013E 7321            JNB     0161
15B6:0140 C60508          MOV     BYTE PTR [DI],08
15B6:0143 47              INC     DI
15B6:0144 47              INC     DI
15B6:0145 47              INC     DI
15B6:0146 BFAC00          MOV     DI,00AC
15B6:0149 47              INC     DI
15B6:014A EB07            JMP     0153
15B6:014C 90              NOP
15B6:014D 90              NOP
15B6:014E 90              NOP
15B6:014F 90              NOP
15B6:0150 90              NOP
15B6:0151 90              NOP
15B6:0152 90              NOP
15B6:0153 BD007E          MOV     BP,7E00
15B6:0156 41              INC     CX
15B6:0157 39D9            CMP     CX,BX
15B6:0159 7414            JZ      016F
15B6:015B D1E8            SHR     AX,1
15B6:015D 7327            JNB     0186
15B6:015F EBF5            JMP     0156
-
```

```
-
-u161 1c0
15B6:0161 BD007C          MOV     BP,7C00
15B6:0164 41              INC     CX
15B6:0165 39D9            CMP     CX,BX
15B6:0167 7406            JZ      016F
15B6:0169 D1E8            SHR     AX,1
15B6:016B 7219            JB      0186
15B6:016D EBF5            JMP     0164
15B6:016F 83C210          ADD     DX,+10
15B6:0172 81FA1008        CMP     DX,0810
15B6:0176 740E            JZ      0186
15B6:0178 8B04            MOV     AX,[SI]
15B6:017A 46              INC     SI
15B6:017B 46              INC     SI
15B6:017C 83C310          ADD     BX,+10
15B6:017F 83FD00          CMP     BP,+00
15B6:0182 74E1            JZ      0165
15B6:0184 EBD1            JMP     0157
15B6:0186 50              PUSH    AX
15B6:0187 56              PUSH    SI
15B6:0188 1E              PUSH    DS
15B6:0189 90              NOP
15B6:018A 51              PUSH    CX
15B6:018B B80030          MOV     AX,3000
15B6:018E 8ED8            MOV     DS,AX
15B6:0190 90              NOP
15B6:0191 89EE            MOV     SI,BP
15B6:0193 83F940          CMP     CX,+40
15B6:0196 7356            JNB     01EE
15B6:0198 01C9            ADD     CX,CX
15B6:019A 01C9            ADD     CX,CX
15B6:019C 01CE            ADD     SI,CX
15B6:019E E80200          CALL    01A3
15B6:01A1 EB77            JMP     021A
15B6:01A3 8A0C            MOV     CL,[SI]
15B6:01A5 46              INC     SI
15B6:01A6 46              INC     SI
15B6:01A7 8B04            MOV     AX,[SI]
15B6:01A9 E80100          CALL    01AD
15B6:01AC C3              RET
15B6:01AD 880D            MOV     [DI],CL
15B6:01AF 47              INC     DI
15B6:01B0 47              INC     DI
15B6:01B1 8905            MOV     [DI],AX
15B6:01B3 47              INC     DI
15B6:01B4 47              INC     DI
15B6:01B5 C3              RET
15B6:01B6 EBAD            JMP     0165
15B6:01B8 90              NOP
15B6:01B9 90              NOP
15B6:01BA 90              NOP
15B6:01BB 59              POP     CX
15B6:01BC 89C8            MOV     AX,CX
15B6:01BE F7F0            DIV     AX
15B6:01C0 80F40F          XOR     AH,0F
-
```

```
u1c2 210
15B6:01C2 0F              DB        0F
15B6:01C3 FEC4            INC       AH
15B6:01C5 88E3            MOV       BL,AH
15B6:01C7 81E3FF00        AND       BX,00FF
15B6:01CB B90100          MOV       CX,0001
15B6:01CE 1F              POP       DS
15B6:01CF 90              NOP
15B6:01D0 5E              POP       SI
15B6:01D1 58              POP       AX
15B6:01D2 81F50002        XOR       BP,0200
15B6:01D6 748D            JZ        0165
15B6:01D8 E97CFF          JMP       0157
15B6:01DB 59              POP       CX
15B6:01DC 29CB            SUB       BX,CX
15B6:01DE B90100          MOV       CX,0001
15B6:01E1 1F              POP       DS
15B6:01E2 90              NOP
15B6:01E3 5E              POP       SI
15B6:01E4 58              POP       AX
15B6:01E5 81F50010        XOR       BP,1000
15B6:01E9 74CB            JZ        01B6
15B6:01EB E969FF          JMP       0157
15B6:01EE 89C8            MOV       AX,CX
15B6:01F0 F7F0            DIV       AX
15B6:01F2 B90001          MOV       CX,0100
15B6:01F5 D0E0            SHL       AL,1
15B6:01F7 D0E0            SHL       AL,1
15B6:01F9 50              PUSH      AX
15B6:01FA 25FF00          AND       AX,00FF
15B6:01FD 01C1            ADD       CX,AX
15B6:01FF 56              PUSH      SI
15B6:0200 01CE            ADD       SI,CX
15B6:0202 E89EFF          CALL      01A3
15B6:0205 5E              POP       SI
15B6:0206 58              POP       AX
15B6:0207 D1E0            SHL       AX,1
15B6:0209 D1E0            SHL       AX,1
15B6:020B 86E0            XCHG      AH,AL
15B6:020D 25FF00          AND       AX,00FF
15B6:0210 01C6            ADD       SI,AX
-u
15B6:0212 E88EFF          CALL      01A3
15B6:0215 EB03            JMP       021A
15B6:0217 90              NOP
15B6:0218 90              NOP
15B6:0219 90              NOP
15B6:021A 81FA1008        CMP       DX,0810
15B6:021E 7403            JZ        0223
15B6:0220 EB8B            JMP       01AD
15B6:0222 90              NOP
15B6:0223 B10C            MOV       CL,0C
15B6:0225 B80008          MOV       AX,0800
15B6:0228 E882FF          CALL      01AD
15B6:022B FF06FEFF        INC       WORD PTR [FFFE]
15B6:022F 893EFEFF        MOV       [FFFE],DI
-
```

```
-u
15B6:0233 E82400          CALL    025A
15B6:0236 813EFEFF7904    CMP     WORD PTR [FFFE],0479
15B6:023C 7207            JB      0245
15B6:023E 744D            JZ      028D
15B6:0240 90              NOP
15B6:0241 90              NOP
15B6:0242 90              NOP
15B6:0243 90              NOP
15B6:0244 90              NOP
15B6:0245 59              POP     CX
15B6:0246 1F              POP     DS
15B6:0247 90              NOP
15B6:0248 5E              POP     SI
15B6:0249 58              POP     AX
15B6:024A 31D2            XOR     DX,DX
15B6:024C BB1000          MOV     BX,0010
15B6:024F 31C9            XOR     CX,CX
15B6:0251 E9DAFE          JMP     012E
-u
15B6:0254 90              NOP
15B6:0255 90              NOP
15B6:0256 90              NOP
15B6:0257 90              NOP
15B6:0258 90              NOP
15B6:0259 90              NOP
15B6:025A BE0080          MOV     SI,8000
15B6:025D 8B3EFCFF        MOV     DI,[FFFC]
15B6:0261 8B0EF8FF        MOV     CX,[FFF8]
15B6:0265 8B2D            MOV     BP,[DI]
15B6:0267 8B14            MOV     DX,[SI]
15B6:0269 46              INC     SI
15B6:026A 46              INC     SI
15B6:026B 8B04            MOV     AX,[SI]
15B6:026D 46              INC     SI
15B6:026E 46              INC     SI
15B6:026F D3C0            ROL     AX,CL
15B6:0271 BBFFFF          MOV     BX,FFFF
-u
15B6:0274 D3E3            SHL     BX,CL
15B6:0276 01D1            ADD     CX,DX
15B6:0278 50              PUSH    AX
15B6:0279 21D8            AND     AX,BX
15B6:027B 81F3FFFF        XOR     BX,FFFF
15B6:027F 89EA            MOV     DX,BP
15B6:0281 21DA            AND     DX,BX
15B6:0283 80F910          CMP     CL,10
15B6:0286 7355            JNB     02DD
15B6:0288 89C5            MOV     BP,AX
15B6:028A 58              POP     AX
15B6:028B EB1F            JMP     02AC
15B6:028D E9D800          JMP     0368
15B6:0290 90              NOP
15B6:0291 90              NOP
15B6:0292 8B16FAFF        MOV     DX,[FFFA]
-
```

```
-u
15B6:0296 81C20010        ADD     DX,1000
15B6:029A 81FA0090        CMP     DX,9000
15B6:029E 74FE            JZ      029E
15B6:02A0 8916FAFF        MOV     [FFFA],DX
15B6:02A4 C3              RET
15B6:02A5 90              NOP
15B6:02A6 90              NOP
15B6:02A7 90              NOP
15B6:02A8 90              NOP
15B6:02A9 90              NOP
15B6:02AA 90              NOP
15B6:02AB 90              NOP
15B6:02AC A1F6FF          MOV     AX,[FFF6]
15B6:02AF 89F2            MOV     DX,SI
15B6:02B1 39C2            CMP     DX,AX
15B6:02B3 75B2            JNZ     0267
15B6:02B5 89E8            MOV     AX,BP
-u
15B6:02B7 E81000          CALL    02CA
15B6:02BA 89FA            MOV     DX,DI
15B6:02BC 8916FCFF        MOV     [FFFC],DX
15B6:02C0 890EF8FF        MOV     [FFF8],CX
15B6:02C4 C3              RET
15B6:02C5 90              NOP
15B6:02C6 90              NOP
15B6:02C7 90              NOP
15B6:02C8 90              NOP
15B6:02C9 90              NOP
15B6:02CA 1E              PUSH    DS
15B6:02CB 90              NOP
15B6:02CC 8B16FAFF        MOV     DX,[FFFA]
15B6:02D0 8EDA            MOV     DS,DX
15B6:02D2 90              NOP
15B6:02D3 8905            MOV     [DI],AX
15B6:02D5 1F              POP     DS
15B6:02D6 90              NOP
-u
15B6:02D7 C3              RET
15B6:02D8 90              NOP
15B6:02D9 90              NOP
15B6:02DA 90              NOP
15B6:02DB 90              NOP
15B6:02DC 90              NOP
15B6:02DD 1E              PUSH    DS
15B6:02DE 90              NOP
15B6:02DF 8B16FAFF        MOV     DX,[FFFA]
15B6:02E3 8EDA            MOV     DS,DX
15B6:02E5 90              NOP
15B6:02E6 8905            MOV     [DI],AX
15B6:02E8 1F              POP     DS
15B6:02E9 90              NOP
15B6:02EA 47              INC     DI
15B6:02EB 47              INC     DI
15B6:02EC 7425            JZ      0313
15B6:02EE C3              RET
15B6:02EF 90              NOP
15B6:02F0 90              NOP
15B6:02F1 90              NOP
15B6:02F2 90              NOP
15B6:02F3 81F3FFFF        XOR     BX,FFFF
-
```

```
-u
15B6:02F7 58              POP     AX
15B6:02F8 21D8            AND     AX,BX
15B6:02FA 89C5            MOV     BP,AX
15B6:02FC 89C8            MOV     AX,CX
15B6:02FE 2D1000          SUB     AX,0010
15B6:0301 89C1            MOV     CX,AX
15B6:0303 E8A6FF          CALL    02AC
15B6:0306 EBD5            JMP     02DD
15B6:0308 90              NOP
15B6:0309 90              NOP
15B6:030A 90              NOP
15B6:030B 90              NOP
15B6:030C 90              NOP
15B6:030D 90              NOP
15B6:030E 90              NOP
15B6:030F 90              NOP
15B6:0310 90              NOP
15B6:0311 90              NOP
15B6:0312 90              NOP
15B6:0313 8B16FAFF        MOV     DX,[FFFA]
-u
15B6:0317 81C30010        ADD     BX,1000
15B6:031B 81FA0090        CMP     DX,9000
15B6:031F 74FE            JZ      031F
15B6:0321 8916FAFF        MOV     [FFFA],DX
15B6:0325 EBCC            JMP     02F3
15B6:0327 90              NOP
15B6:0328 90              NOP
15B6:0329 90              NOP
15B6:032A 90              NOP
15B6:032B 90              NOP
15B6:032C 90              NOP
15B6:032D 90              NOP
15B6:032E 90              NOP
15B6:032F 90              NOP
15B6:0330 B80030          MOV     AX,3000
15B6:0333 8ED8            MOV     DS,AX
15B6:0335 90              NOP
15B6:0336 31C9            XOR     CX,CX
-u
15B6:0338 31F6            XOR     SI,SI
15B6:033A B8FFFF          MOV     AX,FFFF
15B6:033D 8904            MOV     [SI],AX
15B6:033F 46              INC     SI
15B6:0340 41              INC     CX
15B6:0341 31C0            XOR     AX,AX
15B6:0343 8904            MOV     [SI],AX
15B6:0345 46              INC     SI
15B6:0346 41              INC     CX
15B6:0347 81F90001        CMP     CX,0100
15B6:034B 75ED            JNZ     033A
15B6:034D 31C0            XOR     AX,AX
15B6:034F 31F6            XOR     SI,SI
15B6:0351 31C9            XOR     CX,CX
15B6:0353 C3              RET
15B6:0354 90              NOP
15B6:0355 90              NOP
15B6:0356 90              NOP
15B6:0357 90              NOP
-
```

```
-u
15B6:0358 90              NOP
15B6:0359 90              NOP
15B6:035A 90              NOP
15B6:035B 90              NOP
15B6:035C 90              NOP
15B6:035D 90              NOP
15B6:035E 90              NOP
15B6:035F 90              NOP
15B6:0360 90              NOP
15B6:0361 90              NOP
15B6:0362 90              NOP
15B6:0363 90              NOP
15B6:0364 90              NOP
15B6:0365 90              NOP
15B6:0366 90              NOP
15B6:0367 90              NOP
15B6:0368 59              POP       CX
15B6:0369 1F              POP       DS
15B6:036A 90              NOP
15B6:036B 5E              POP       SI
15B6:036C 58              POP       AX
15B6:036D 5A              POP       DX
15B6:036E CD20            INT       20
15B6:0370 90              NOP
15B6:0371 90              NOP
15B6:0372 90              NOP
15B6:0373 90              NOP
15B6:0374 90              NOP
15B6:0375 90              NOP
15B6:0376 90              NOP
15B6:0377 90              NOP
-u
15B6:0378 90              NOP
15B6:0379 90              NOP
15B6:037A 90              NOP
15B6:037B 90              NOP
15B6:037C 90              NOP
15B6:037D 90              NOP
15B6:037E 90              NOP
15B6:037F 90              NOP
15B6:0380 90              NOP
15B6:0381 0900            OR        [BX+SI],AX
15B6:0383 EB03            JMP       0388
15B6:0385 BA4C98          MOV       DX,984C
15B6:0388 E8A92B          CALL      2F34
15B6:038B C3              RET
15B6:038C C606099701      MOV       BYTE PTR [9709],01
15B6:0391 89160BA6        MOV       [A60B],DX
15B6:0395 C6060D9701      MOV       BYTE PTR [970D],01
-
```

Apêndice B - Listagem completa do Programa de Decodificação

```
-u100
15B6:0100 31C0          XOR     AX,AX
15B6:0102 31DB          XOR     BX,BX
15B6:0104 31C9          XOR     CX,CX
15B6:0106 31D2          XOR     DX,DX
15B6:0108 31FF          XOR     DI,DI
15B6:010A 31ED          XOR     BP,BP
15B6:010C 31F6          XOR     SI,SI
15B6:010E BA0040        MOV     DX,4000
15B6:0111 8EC2          MOV     ES,DX
15B6:0113 BA0030        MOV     DX,3000
15B6:0116 8EDA          MOV     DS,DX
15B6:0118 90            NOP
15B6:0119 A30015        MOV     [1500],AX
15B6:011C A30815        MOV     [1508],AX
15B6:011F A32015        MOV     [1520],AX
-u
15B6:0122 A34015        MOV     [1540],AX
15B6:0125 B80068        MOV     AX,6800
15B6:0128 89C6          MOV     SI,AX
15B6:012A A31015        MOV     [1510],AX
15B6:012D C7040000      MOV     WORD PTR [SI],0000
15B6:0131 B80060        MOV     AX,6000
15B6:0134 A32615        MOV     [1526],AX
15B6:0137 B80F00        MOV     AX,000F
15B6:013A A33015        MOV     [1530],AX
15B6:013D B80400        MOV     AX,0004
15B6:0140 A33415        MOV     [1534],AX
-u
15B6:0143 B80040        MOV     AX,4000
15B6:0146 A32215        MOV     [1522],AX
15B6:0149 B80020        MOV     AX,2000
15B6:014C 89C5          MOV     BP,AX
15B6:014E BB0020        MOV     BX,2000
15B6:0151 BE0068        MOV     SI,6800
15B6:0154 C7040000      MOV     WORD PTR [SI],0000
15B6:0158 BE0000        MOV     SI,0000
15B6:015B 8CC0          MOV     AX,ES
15B6:015D 50            PUSH    AX
15B6:015E 25FF0F        AND     AX,0FFF
15B6:0161 3D0008        CMP     AX,0800
-
```

```
-u
15B6:0164 750A          JNZ     0170
15B6:0166 B10C          MOV     CL,0C
15B6:0168 58            POP     AX
15B6:0169 D3E8          SHR     AX,CL
15B6:016B A33815        MOV     [1538],AX
15B6:016E EB05          JMP     0175
15B6:0170 E98D02        JMP     0400
15B6:0173 90            NOP
15B6:0174 90            NOP
15B6:0175 89E8          MOV     AX,BP
15B6:0177 0304          ADD     AX,[SI]
15B6:0179 89C6          MOV     SI,AX
15B6:017B 8B04          MOV     AX,[SI]
15B6:017D D0E8          SHR     AL,1
15B6:017F 7415          JZ      0196
15B6:0181 7319          JNB     019C
15B6:0183 251F00        AND     AX,001F
-
```

```
u186 1e7
15B6:0186 B106          MOV     CL,06
15B6:0188 D3E0          SHL     AX,CL
15B6:018A A31015        MOV     [1510],AX
15B6:018D 90            NOP
15B6:018E 89E8          MOV     AX,BP
15B6:0190 01C6          ADD     SI,AX
15B6:0192 8B04          MOV     AX,[SI]
15B6:0194 EB3A          JMP     01D0
15B6:0196 E9AC00        JMP     0245
15B6:0199 90            NOP
15B6:019A 90            NOP
15B6:019B 90            NOP
15B6:019C 253F00        AND     AX,003F
15B6:019F 8B361015      MOV     SI,[1510]
15B6:01A3 0304          ADD     AX,[SI]
15B6:01A5 8904          MOV     [SI],AX
-u
15B6:01A7 90            NOP
15B6:01A8 8B1E4015      MOV     BX,[1540]
15B6:01AC 01D8          ADD     AX,BX
15B6:01AE A34015        MOV     [1540],AX
15B6:01B1 3D0008        CMP     AX,0800
15B6:01B4 7414          JZ      01CA
15B6:01B6 46            INC     SI
15B6:01B7 46            INC     SI
15B6:01B8 C7040000      MOV     WORD PTR [SI],0000
15B6:01BC 8B04          MOV     AX,[SI]
15B6:01BE A31015        MOV     [1510],AX
15B6:01C1 89E8          MOV     AX,BP
15B6:01C3 350060        XOR     AX,6000
15B6:01C6 89C5          MOV     BP,AX
-u
15B6:01C8 EB06          JMP     01D0
15B6:01CA E9C301        JMP     0390
15B6:01CD 90            NOP
15B6:01CE 90            NOP
15B6:01CF 90            NOP
15B6:01D0 31C0          XOR     AX,AX
15B6:01D2 A33415        MOV     [1534],AX
15B6:01D5 A33815        MOV     [1538],AX
15B6:01D8 89E8          MOV     AX,BP
15B6:01DA 3D0020        CMP     AX,2000
15B6:01DD 740D          JZ      01EC
15B6:01DF E8A300        CALL    0285
15B6:01E2 E8A000        CALL    0285
15B6:01E5 EB8E          JMP     0175
15B6:01E7 90            NOP
-
```

```
-u
15B6:01E8 90              NOP
15B6:01E9 90              NOP
15B6:01EA 90              NOP
15B6:01EB 90              NOP
15B6:01EC E89600          CALL    0285
15B6:01EF E89300          CALL    0285
15B6:01F2 E89000          CALL    0285
15B6:01F5 E88D00          CALL    0285
15B6:01F8 E97AFF          JMP     0175
15B6:01FB 90              NOP
15B6:01FC 90              NOP
15B6:01FD 90              NOP
15B6:01FE 90              NOP
15B6:01FF 90              NOP
15B6:0200 B80020          MOV     AX,2000
15B6:0203 89C5            MOV     BP,AX
15B6:0205 C70640150000    MOV     WORD PTR [1540],0000
-u
15B6:020B EBC3            JMP     01D0
15B6:020D 90              NOP
15B6:020E 90              NOP
15B6:020F 90              NOP
15B6:0210 90              NOP
15B6:0211 90              NOP
15B6:0212 90              NOP
15B6:0213 90              NOP
15B6:0214 90              NOP
15B6:0215 90              NOP
15B6:0216 BF0060          MOV     DI,6000
15B6:0219 31C0            XOR     AX,AX
15B6:021B 8805            MOV     [DI],AL
15B6:021D 47              INC     DI
15B6:021E 81FF0068        CMP     DI,6800
15B6:0222 75F7            JNZ     021B
15B6:0224 C3              RET
15B6:0225 20060815        AND     [1508],AL
15B6:0229 FF060015        INC     WORD PTR [1500]
-u
15B6:022D A10015          MOV     AX,[1500]
15B6:0230 90              NOP
15B6:0231 3D0800          CMP     AX,0008
15B6:0234 75CA            JNZ     0200
15B6:0236 E89F00          CALL    02D8
15B6:0239 31C0            XOR     AX,AX
15B6:023B A30015          MOV     [1500],AX
15B6:023E EBC0            JMP     0200
15B6:0240 90              NOP
15B6:0241 90              NOP
15B6:0242 90              NOP
15B6:0243 90              NOP
15B6:0244 90              NOP
15B6:0245 A13415          MOV     AX,[1534]
15B6:0248 3D0D00          CMP     AX,000D
15B6:024B 74FE            JZ      024B
-
```

```
-u
15B6:024D E83500          CALL    0285
15B6:0250 A13415          MOV     AX,[1534]
15B6:0253 3D0C00          CMP     AX,000C
15B6:0256 7408            JZ      0260
15B6:0258 E91AFF          JMP     0175
15B6:025B 90              NOP
15B6:025C 90              NOP
15B6:025D 90              NOP
15B6:025E 90              NOP
15B6:025F 90              NOP
15B6:0260 A13815          MOV     AX,[1538]
15B6:0263 90              NOP
15B6:0264 25FF0F          AND     AX,0FFF
15B6:0267 3D0100          CMP     AX,0001
15B6:026A 74B9            JZ      0225
15B6:026C E906FF          JMP     0175
-u
15B6:026F 06              PUSH    ES
15B6:0270 40              INC     AX
15B6:0271 15813E          ADC     AX,3E81
15B6:0274 40              INC     AX
15B6:0275 151008          ADC     AX,0810
15B6:0278 7486            JZ      0200
15B6:027A E9F8FE          JMP     0175
15B6:027D 90              NOP
15B6:027E 90              NOP
15B6:027F 90              NOP
15B6:0280 90              NOP
15B6:0281 90              NOP
15B6:0282 90              NOP
15B6:0283 90              NOP
15B6:0284 90              NOP
15B6:0285 FE063015        INC     BYTE PTR [1530]
15B6:0289 803E301510      CMP     BYTE PTR [1530],10
15B6:028E 7216            JB      02A6
-u
15B6:0290 B000            MOV     AL,00
15B6:0292 A23015          MOV     [1530],AL
15B6:0295 FF062015        INC     WORD PTR [1520]
15B6:0299 FF062015        INC     WORD PTR [1520]
15B6:029D 7507            JNZ     02A6
15B6:029F 8CC0            MOV     AX,ES
15B6:02A1 050010          ADD     AX,1000
15B6:02A4 8EC0            MOV     ES,AX
15B6:02A6 FE063415        INC     BYTE PTR [1534]
15B6:02AA 8B362015        MOV     SI,[1520]
15B6:02AE B80100          MOV     AX,0001
-
```

```
-u
15B6:02B1 8A0E3015      MOV     CL,[1530]
15B6:02B5 D3E0          SHL     AX,CL
15B6:02B7 2304          AND     AX,[SI]
15B6:02B9 7505          JNZ     02C0
15B6:02BB C3            RET
15B6:02BC 90            NOP
15B6:02BD 90            NOP
15B6:02BE 90            NOP
15B6:02BF 90            NOP
15B6:02C0 8A0E3415      MOV     CL,[1534]
15B6:02C4 FEC9          DEC     CL
15B6:02C6 B80100        MOV     AX,0001
15B6:02C9 D3E0          SHL     AX,CL
15B6:02CB 09063815      OR      [1538],AX
15B6:02CF C3            RET
15B6:02D0 90            NOP
-u
15B6:02D1 90            NOP
15B6:02D2 90            NOP
15B6:02D3 90            NOP
15B6:02D4 90            NOP
15B6:02D5 90            NOP
15B6:02D6 90            NOP
15B6:02D7 90            NOP
15B6:02D8 E83BFF        CALL    0216
15B6:02DB B380          MOV     BL,80
15B6:02DD BE0068        MOV     SI,6800
15B6:02E0 BF0060        MOV     DI,6000
15B6:02E3 8B04          MOV     AX,[SI]
15B6:02E5 46            INC     SI
15B6:02E6 46            INC     SI
15B6:02E7 01C7          ADD     DI,AX
15B6:02E9 81FFFF67      CMP     DI,67FF
15B6:02ED 7713          JA      0302
15B6:02EF 8B0C          MOV     CX,[SI]
-u
15B6:02F1 46            INC     SI
15B6:02F2 46            INC     SI
15B6:02F3 081D          OR      [DI],BL
15B6:02F5 47            INC     DI
15B6:02F6 81FFFF67      CMP     DI,67FF
15B6:02FA 7706          JA      0302
15B6:02FC 49            DEC     CX
15B6:02FD 75F4          JNZ     02F3
15B6:02FF EBE2          JMP     02E3
15B6:0301 90            NOP
15B6:0302 D0EB          SHR     BL,1
15B6:0304 73DA          JNB     02E0
15B6:0306 E80700        CALL    0310
15B6:0309 C3            RET
15B6:030A 90            NOP
15B6:030B 90            NOP
15B6:030C 90            NOP
15B6:030D 90            NOP
15B6:030E 90            NOP
15B6:030F 90            NOP
15B6:0310 BA0800        MOV     DX,0008
-
```

```
-u
15B6:0313 E81A00          CALL    0330
15B6:0316 B405            MOV     AH,05
15B6:0318 B20A            MOV     DL,0A
15B6:031A CD21            INT     21
15B6:031C B405            MOV     AH,05
15B6:031E B20A            MOV     DL,0A
15B6:0320 CD21            INT     21
15B6:0322 B405            MOV     AH,05
15B6:0324 B20A            MOV     DL,0A
15B6:0326 CD21            INT     21
15B6:0328 4A              DEC     DX
15B6:0329 75E8            JNZ     0313
15B6:032B C3              RET
15B6:032C 90              NOP
15B6:032D 90              NOP
15B6:032E 90              NOP
15B6:032F 90              NOP
15B6:0330 B405            MOV     AH,05
15B6:0332 B21B            MOV     DL,1B
-u
15B6:0334 CD21            INT     21
15B6:0336 B405            MOV     AH,05
15B6:0338 B240            MOV     DL,40
15B6:033A CD21            INT     21
15B6:033C B405            MOV     AH,05
15B6:033E B21B            MOV     DL,1B
15B6:0340 CD21            INT     21
15B6:0342 B405            MOV     AH,05
15B6:0344 B24B            MOV     DL,4B
15B6:0346 CD21            INT     21
15B6:0348 B405            MOV     AH,05
15B6:034A B200            MOV     DL,00
15B6:034C CD21            INT     21
15B6:034E B405            MOV     AH,05
15B6:0350 B203            MOV     DL,03
15B6:0352 CD21            INT     21
-u
15B6:0354 BE0060          MOV     SI,6000
15B6:0357 BB0063          MOV     BX,6300
15B6:035A B405            MOV     AH,05
15B6:035C 8A14            MOV     DL,[SI]
15B6:035E CD21            INT     21
15B6:0360 46              INC     SI
15B6:0361 39DE            CMP     SI,BX
15B6:0363 75F5            JNZ     035A
15B6:0365 B405            MOV     AH,05
15B6:0367 B21B            MOV     DL,1B
15B6:0369 CD21            INT     21
15B6:036B B405            MOV     AH,05
15B6:036D B233            MOV     DL,33
15B6:036F CD21            INT     21
15B6:0371 B405            MOV     AH,05
15B6:0373 B208            MOV     DL,08
-
```

```
-u
15B6:0437 CD21            INT       21
15B6:0439 B409            MOV       AH,09
15B6:043B B24E            MOV       DL,4E
15B6:043D CD21            INT       21
15B6:043F B409            MOV       AH,09
15B6:0441 B253            MOV       DL,53
15B6:0443 CD21            INT       21
15B6:0445 B409            MOV       AH,09
15B6:0447 B24D            MOV       DL,4D
15B6:0449 CD21            INT       21
15B6:044B B409            MOV       AH,09
15B6:044D B249            MOV       DL,49
15B6:044F CD21            INT       21
15B6:0451 B409            MOV       AH,09
15B6:0453 B253            MOV       DL,53
15B6:0455 CD21            INT       21
-u
15B6:0457 B409            MOV       AH,09
15B6:0459 B253            MOV       DL,53
15B6:045B CD21            INT       21
15B6:045D B409            MOV       AH,09
15B6:045F B241            MOV       DL,41
15B6:0461 CD21            INT       21
15B6:0463 B409            MOV       AH,09
15B6:0465 B24F            MOV       DL,4F
15B6:0467 CD21            INT       21
15B6:0469 B409            MOV       AH,09
15B6:046B B220            MOV       DL,20
15B6:046D CD21            INT       21
15B6:046F B409            MOV       AH,09
15B6:0471 B220            MOV       DL,20
15B6:0473 CD21            INT       21
15B6:0475 B409            MOV       AH,09
-u
15B6:0477 B220            MOV       DL,20
15B6:0479 CD21            INT       21
15B6:047B B409            MOV       AH,09
15B6:047D B252            MOV       DL,52
15B6:047F CD21            INT       21
15B6:0481 B409            MOV       AH,09
15B6:0483 B255            MOV       DL,55
15B6:0485 CD21            INT       21
15B6:0487 B409            MOV       AH,09
15B6:0489 B249            MOV       DL,49
15B6:048B CD21            INT       21
15B6:048D B409            MOV       AH,09
15B6:048F B24D            MOV       DL,4D
15B6:0491 CD21            INT       21
15B6:0493 4D              DEC       BP
15B6:0494 75FE            JNZ       0494
15B6:0496 90              NOP
-
```

```
-u
15B6:03D5 A13415          MOV     AX,[1534]
15B6:03D8 3D0C00          CMP     AX,000C
15B6:03DB 74FE            JZ      03DB
15B6:03DD EBFE            JMP     03DD
15B6:03DF 8CD8            MOV     AX,DS
15B6:03E1 90              NOP
15B6:03E2 25FF0F          AND     AX,0FFF
15B6:03E5 3D0100          CMP     AX,0001
15B6:03E8 74FE            JZ      03E8
15B6:03EA EBFE            JMP     03EA
15B6:03EC FF064015        INC     WORD PTR [1540]
15B6:03F0 813E40150008    CMP     WORD PTR [1540],0800
-u
15B6:03F6 74FE            JZ      03F6
15B6:03F8 EBFE            JMP     03F8
15B6:03FA C3              RET
15B6:03FB 90              NOP
15B6:03FC 90              NOP
15B6:03FD 90              NOP
15B6:03FE 90              NOP
15B6:03FF 90              NOP
15B6:0400 A13815          MOV     AX,[1538]
15B6:0403 3D0010          CMP     AX,1000
15B6:0406 74FE            JZ      0406
15B6:0408 21C0            AND     AX,AX
15B6:040A 74FE            JZ      040A
15B6:040C B80F00          MOV     AX,000F
15B6:040F 89C5            MOV     BP,AX
15B6:0411 B90000          MOV     CX,0000
15B6:0414 BA4F1E          MOV     DX,1E4F
-u
15B6:0417 B707            MOV     BH,07
15B6:0419 B80006          MOV     AX,0600
15B6:041C CD21            INT     21
15B6:041E BA0C00          MOV     DX,000C
15B6:0421 B71D            MOV     BH,1D
15B6:0423 B402            MOV     AH,02
15B6:0425 CD21            INT     21
15B6:0427 B409            MOV     AH,09
15B6:0429 B254            MOV     DL,54
15B6:042B CD21            INT     21
15B6:042D B409            MOV     AH,09
15B6:042F B252            MOV     DL,52
15B6:0431 CD21            INT     21
15B6:0433 B409            MOV     AH,09
15B6:0435 B241            MOV     DL,41
-
```

```
-u
15B6:0437 CD21            INT       21
15B6:0439 B409            MOV       AH,09
15B6:043B B24E            MOV       DL,4E
15B6:043D CD21            INT       21
15B6:043F B409            MOV       AH,09
15B6:0441 B253            MOV       DL,53
15B6:0443 CD21            INT       21
15B6:0445 B409            MOV       AH,09
15B6:0447 B24D            MOV       DL,4D
15B6:0449 CD21            INT       21
15B6:044B B409            MOV       AH,09
15B6:044D B249            MOV       DL,49
15B6:044F CD21            INT       21
15B6:0451 B409            MOV       AH,09
15B6:0453 B253            MOV       DL,53
15B6:0455 CD21            INT       21
-u
15B6:0457 B409            MOV       AH,09
15B6:0459 B253            MOV       DL,53
15B6:045B CD21            INT       21
15B6:045D B409            MOV       AH,09
15B6:045F B241            MOV       DL,41
15B6:0461 CD21            INT       21
15B6:0463 B409            MOV       AH,09
15B6:0465 B24F            MOV       DL,4F
15B6:0467 CD21            INT       21
15B6:0469 B409            MOV       AH,09
15B6:046B B220            MOV       DL,20
15B6:046D CD21            INT       21
15B6:046F B409            MOV       AH,09
15B6:0471 B220            MOV       DL,20
15B6:0473 CD21            INT       21
15B6:0475 B409            MOV       AH,09
-u
15B6:0477 B220            MOV       DL,20
15B6:0479 CD21            INT       21
15B6:047B B409            MOV       AH,09
15B6:047D B252            MOV       DL,52
15B6:047F CD21            INT       21
15B6:0481 B409            MOV       AH,09
15B6:0483 B255            MOV       DL,55
15B6:0485 CD21            INT       21
15B6:0487 B409            MOV       AH,09
15B6:0489 B249            MOV       DL,49
15B6:048B CD21            INT       21
15B6:048D B409            MOV       AH,09
15B6:048F B24D            MOV       DL,4D
15B6:0491 CD21            INT       21
15B6:0493 4D              DEC       BP
15B6:0494 75FE            JNZ       0494
15B6:0496 90              NOP
-
```

```
-u
15B6:0497 90              NOP
15B6:0498 90              NOP
15B6:0499 90              NOP
15B6:049A 90              NOP
15B6:049B 90              NOP
15B6:049C 90              NOP
15B6:049D 90              NOP
15B6:049E 90              NOP
15B6:049F 90              NOP
15B6:04A0 D1ED            SHR       BP,1
15B6:04A2 4A              DEC       DX
15B6:04A3 74C5            JZ        046A
15B6:04A5 D1D3            RCL       BX,1
15B6:04A7 D1ED            SHR       BP,1
15B6:04A9 4A              DEC       DX
15B6:04AA 74C4            JZ        0470
15B6:04AC D1D3            RCL       BX,1
15B6:04AE 85DB            TEST      BX,BX
15B6:04B0 7417            JZ        04C9
15B6:04B2 D1ED            SHR       BP,1
15B6:04B4 4A              DEC       DX
15B6:04B5 74BF            JZ        0476
-u
15B6:04B7 D1D3            RCL       BX,1
15B6:04B9 80FB06          CMP       BL,06
15B6:04BC 720B            JB        04C9
15B6:04BE D1ED            SHR       BP,1
15B6:04C0 4A              DEC       DX
15B6:04C1 7504            JNZ       04C7
15B6:04C3 AD              LODSW
15B6:04C4 95              XCHG      BP,AX
15B6:04C5 B210            MOV       DL,10
15B6:04C7 D1D3            RCL       BX,1
15B6:04C9 2E              CS:
15B6:04CA 8A8F6101        MOV       CL,[BX+0161]
15B6:04CE 80F90A          CMP       CL,0A
15B6:04D1 7475            JZ        0548
15B6:04D3 33DB            XOR       BX,BX
15B6:04D5 83F902          CMP       CX,+02
-
```

Apêndice C - Listagem completa do Programa HUFCOD em BASIEX

```
10  ' ***************************************************
15  ' PROGRAMA PARA COLOCAR CODIGO DE HUFFMAN NA MEMORIA
20  ' (C) FEV/ 1991          SET/1994
25  ' ***************************************************
30  ' *              PROJETO FAX-PC                     *
35  ' ***************************************************
40 GOSUB 7000
50 ON KEY(1) GOSUB 1800:'F1 - VOLTA NO CODIGOS
55 ON KEY(2) GOSUB 1850:'F2 - AVANCA
60 ON KEY(3) GOSUB 1900:'F3 - NOVO COMPRIMENTO
65 ON KEY(4) GOSUB 1950:'F4 - MENU
70 ON KEY(5) GOSUB 5000:'F5 - ENTRAR COM CODIGOS
100 CLS
110 PRINT" PROGRAMA PARA COLOCAR CODIGOS DE HUFFMAN"
120 PRINT:PRINT:PRINT"TECLE OPAO Desejada:"
130 PRINT"     1 - ENTRAR COM CODIGOS "
140 PRINT"     2 - SALVAR EM DISQUETE "
150 PRINT"     3 - CARREGAR NA MEMORIA "
160 PRINT"     4 - SAI DO PROGRAMA      "
170 INPUT" OPAO    ";A
180 IF A<1 OR A>5 THEN GOTO 100
190 ON A GOTO 1000, 2000, 3000, 4000
1000 '*****************************************
1001 ' Rotina para entrar com codigos de Huffman
1005 '*****************************************
1010 CLS
1020 PRINT"  ENTRADA DOS CODIGOS "
1030 PRINT: INPUT" QUAL O COMPRIMENTO DA CADEIA ";T
1040 IF T>=0 AND T<=2048 THEN 1060
1050 PRINT:PRINT" FORA DA FAIXA PERMITIDA..... REDIGITE":GOTO 1030
1060 IF T<64 THEN 1080
1070 IF (T/64)= INT(T/64) THEN 1080
1075 PRINT:PRINT"COMPRIMENTO DE CADEIA INEXISTENTE..... REDIGITE":GOTO 1030
1080 PRINT:PRINT" F1 - VOLTA  F2 - AVANA  F3 - NOVO COMPRIMENTO  F4 - MENU
- ENTRA"
1085 PRINT:PRINT"COMPRIMENTO";TAB(20); "CODIGO"; TAB(36); "CODIGO"
1090 PRINT" DA CADEIA "; TAB(20); "BRANCA"; TAB(36); "PRETA"
1100 FOR A=1 TO 5
1110 KEY(A) ON
1120 NEXT A
1130 GOSUB 6000
1140 GOTO 1140
1800 '*************************
1805 ' KEY F1 - Volta
1810 '*************************
1820 IF T<=64 THEN T=T-1 ELSE T=T-64
1830 IF T<0 THEN T=0
1840 RETURN 1130
1850 '*********************************
1855 ' KEY F2 - Avanca
1860 '*********************************
1870 IF T<64 THEN T=T+1 ELSE T=T+64
1880 IF T>2048 THEN T=2048
1890 RETURN 1130
1900 '*********************************
1910 ' KEY F3 - Novo comprimento
1920 '*********************************
1930 RETURN 1030
1950 '*********************************
1960 ' KEY F4 - Menu
1970 '*********************************
1980 RETURN 100
```

```
2000 '*****************************************
2001 ' Salva o dicionario no disquete
2002 '*****************************************
2010 CLS
2020 PRINT"   SALVANDO DICIONARIO NO DISQUETE"
2030 PRINT:PRINT:INPUT "NOME DO ARQUIVO ";N$
2040 DEF SEG=&H3000
2050 BSAVE N$,&H7C00,&H3FF
2060 GOTO 100
3000 '*************************************
3001 'Carrega na memoria o dicionario
3002 '*************************************
3010 CLS
3020 PRINT"      CARREGANDO DICIONARIO NA MEMORIA"
3030 PRINT:PRINT:PRINT:PRINT:INPUT "NOME DO ARQUIVO";N$
3040 DEF SEG=&H3000
3050 BLOAD N$,&H7C00
3060 GOTO 100
4000 '**********************
4001 ' Saida do programa
4002 '**********************
4009 PRINT:INPUT"Ques salvar as mudancas? (S/N)";A$
4010 IF A$="N" THEN 4060
4020 PRINT" Salve o programa atraves de F4"
4030 PRINT:INPUT"Tem certeza que quer terminar o programa? (S/N)";R$
4040 IF R$="N" THEN 100
4042 FOR I=1 TO 500:PRINT:NEXT I
4050 PRINT" Aviso: Saindo do programa para Sistema"
4052 FOR I=1 TO 500:PRINT:NEXT I
4060 SYSTEM
5000 '**************************
5001 ' Entra com o codigos
5002 '**************************
5009 INPUT"Branca ";B$
5010 X$=B$
5020 GOSUB 6700
5030 IF V$="T" THEN 5060
5040     PRINT" Codigo nao reconhecido"
5050     GOTO 5000
5060 GOSUB 6900
5070 ENDE=&H7C00
5080 GOSUB 6500
5090 INPUT " Preto ";P$
5100 X$=P$
5110 GOSUB 6700
5120 IF V$="T" THEN 5150
5130           PRINT" CODIGO NAO RECONHECIDO"
5140           GOTO 5000
5150 GOSUB 6900
5160 ENDE=&H7E00
5170 GOSUB 6500
5180 GOSUB 6000
5190 IF T<64 THEN T=T+1 ELSE T=T+64
5192 IF T>2048 THEN T=2048
5200 GOSUB 6000
5210 GOTO 5000
```

```
6000 '*******************************
6001 ' SUBROTINA MOSTRA
6002 '*******************************
6010 DEF SEG=&H3000
6020 PRINT TAB(4); T ; TAB(23);
6030 ENDE=&H7C00
6040 GOSUB 6080
6050 PRINT TAB(38);
6060 ENDE=&H7E00
6065 GOSUB 6080
6070 PRINT:RETURN
6080 '*************************
6081 ' Sub-rotina composicao ta mais bonito!!
6082 '*************************
6085 IF T>63 THEN X=INT(T/64)+63 ELSE X=T
6090 ENDE=ENDE+(X*4)
6100 V  = PEEK(ENDE)
6110 C  = PEEK(ENDE+1)
6120 C1 = PEEK(ENDE+2)
6130 C2 = PEEK(ENDE+3)
6140 CX = C1 + 256*C2
6150 IF CX>16384 OR V<1 OR V>15 THEN 6270
6190 FOR XX=1 TO V
6200     RT=CX-2*INT(CX/2)
6210     IF RT=1 THEN PRINT"1"; ELSE PRINT"0";
6220     CX=INT(CX/2)
6230 NEXT XX
6240 GOTO 6285
6270 ' Erro na sub-rotina'
6280 PRINT "Erro";
6285 ' Saida da sub-rotina
6290 RETURN
6500 '*************************
6501 '    Subrotina armazena
6502 '*************************
6510 IF T>63 THEN X=INT(T/64)+63 ELSE X=T
6520 DEF SEG=&H3000
6530 ENDE=ENDE+(X*4)
6540 POKE ENDE,LEN(X$)
6550 POKE ENDE+1,0
6560 POKE ENDE+2,X1
6570 POKE ENDE+3,X2
6580 RETURN


6700 '*************************
6701 ' Subrotina verifica
6702 '*************************
6703 V$="T"
6710 V=LEN(X$)
6720 IF V<2 OR V>14 THEN V$="F":GOTO 6780
6730 FOR XX=1 TO V
6740     K$=MID$(X$,XX,1)
6750     IF K$<>"0" AND K$<>"1" THEN V$="F":XX=V
6760 NEXT XX
6780 RETURN
```

```
6900 '*****************************
6901 ' Subrotina Convert
6902 '*****************************
6910 B=1
6920 N=0
6930 FOR XX=1 TO V
6940      IF MID$(X$,XX,1)="1" THEN N=N+B
6950      B=B+B
6960 NEXT XX
6970 X2=INT(N/256)
6980 X1=N-256*X2
6990 RETURN
7000 '*****************************
7010 '     Subrotina zera memoria
7020 '*****************************
7030 DEF SEG=&H3000
7035 ENDE=&H7C00
7040 FOR I=0 TO 1023
7050      ZERA=ENDE+I
7060      POKE ZERA,0
7070 NEXT I
7080 RETURN
```